SMC in Estimation of a State Space Model

Dong-Whan Ko
Department of Economics
Rutgers, the State University of New Jersey
December 31, 2012

Abstract

I briefly summarize procedures for macroeconomic Dynamic Stochastic General Equilibrium (DSGE) model estimation. I then review the basic Sequential Monte Carlo (SMC) methods. Especially, I review Bootstrap Filter and Auxiliary Particle Filter for state filtering method and Liu and West filter and Storvik Filter for parameter learning as a building block for Particle Learning. I construct simple linear state space model. I then obtain the likelihood using different filtering method: Kalman filter and Auxiliary Particle Filter. I also estimate the parameters using Markov Chain Monte Carlo (MCMC) and Particle Learning. Comparing the results, I confirm that SMC method can be alternative to MCMC method for DSGE model estimation.

1 Introduction

In Dynamic Stochastic General Equilibrium (DSGE) model, we have a system of non-linear first order conditions. In order to solve the model, researchers usually log-linearize the model and construct linear Gaussian state space model. A typical way to estimate the structural parameters is to calculate likelihood with Kalman filter at first and then estimate the structural parameters with random walk Metropolis-Hasting algorithm. Rubio-Ramirez and Fernandez-Villaverde (2005), however, tried to solve the system of equations non-linearly with finite-element method and then applied the particle filters to obtain the likelihood. They argued that for both of simulated and real data, particle filter method delivers a substantially better fit of the model to the data. Among many sampling methods of the proposal distribution, Rubio-Ramirez and Fernandez-Villaverde (2005) use
the bootstrap filter proposed by Gordon et al. (1993) in which they chose the transition kernel as the proposal distribution, accordingly the weight is simplified to the likelihood at time $t$.\footnote{Details on Particle filter method as one of the Sequential Monte Carlo (SMC) methods are well explained in Doucet and Johansen (2009)} This is not an optimal choice for the proposal, however, because the information of current observation is not used for proposal density, as Creal (2012) pointed out. In addition, Rubio-Ramirez and Fernandez-Villaverde (2005) used particle filter not for estimating the parameters but for calculating the likelihood: they used MCMC method for estimation. On the other hand, Carvalho et al. (2010) present particle learning (hereafter PL) in which one directly samples from the particle approximation to the joint posterior distribution of states and conditional sufficient statistics for fixed parameters in a fully-adapted resample-propagate framework. I will give more detail later.

Chen, Petralia and Lopes (2010) argue that even for a case in which MCMC method is available, the particle filter have several advantages. First, MCMC methods rely on Markov chain convergence which might not be geometrically ergodic for structural models such as DSGE models (Papaspiliopoulos and Roberts (2008)), whereas consistency and asymptotic normality of particle filters have been proven in Douc et al. (2007). Second, when we use particle filter, posterior approximations of the parameters and states can be obtained in on-line manner, that is, it can be calculated at each time period. To obtain the same amount of information with MCMC, one would have to resort to repeated implementation of MCMC at each time period, which is more inefficient in terms of running time and computing resources.

In line with this argument, I construct simple linear Gaussian state space model and see whether SMC can compete with MCMC in estimation of the parameters. For the sake of comparison, I obtain the likelihood and estimates of parameters using different methodologies; Kalman Filter and Auxiliary Particle Filter for calculating the likelihood; Random Walk Metropolis-Hasting algorithm and Particle Learning for the estimates. A general form of state space model is given by

$$ y_t = f(x_t, \theta) \quad \text{where} \quad \nu_t \sim N(0, \Sigma_{\nu}) $$

$$ x_t = g(x_{t-1}, \theta) \quad \text{where} \quad w_t \sim N(0, \Sigma_{w}) $$

(1.1)

for any given $t = 1, \ldots, T$. Where $f(\cdot, \cdot)$ and $g(\cdot, \cdot)$ could be non-linear functions of state variables and the parameters. In addition, it could be including time varying parameters. However, I as-
sume for my example that those are linear functions with fixed parameters. The distribution of observations and state variables is determined by the shocks, \( \nu_t \) and \( w_t \), and the corresponding densities can be written as \( p(y_t|x_t, \theta) \) and \( p(x_t|x_{t-1}, \theta) \) respectively. The sequence of state variables\(^2\), \( x^t = (x_0, ..., x_t) \) and parameters, \( \theta \), are unobservable, thus it has to be estimated using observations. For this, there are three important densities: the one step ahead predictive distribution, \( p(x_t|y^{t-1}; \theta) \), the filtering distribution, \( p(x_t|y^t; \theta) \), and the smoothing distribution, \( p(x_t|y^T; \theta) \).

Generally speaking, given initial distribution of the state variable, one step ahead predictive distribution for state variable, \( p(x_0; \theta) \), can be obtained with Chapman-Kolmogorov Equation:

\[
p(x_t|y^{t-1}; \theta) = \int p(x_t|x_{t-1}; \theta) p(x_{t-1}|y^{t-1}; \theta) \, dx_{t-1}
\]

Updating with observation at \( t \), \( y_t \), Bayes’ theorem gives the filtering distribution:

\[
p(x_t|y^t; \theta) = \frac{p(y_t|x_t|y^{t-1}; \theta)}{p(y_t|y^{t-1}; \theta)} = \frac{p(y_t|x_t, y^{t-1}; \theta) p(x_t|y^{t-1}; \theta)}{p(y_t|y^{t-1}; \theta)}
\]

Since we assume Markovian process in observation and state variables, finally we can get

\[
p(x_t|y^t; \theta) = \frac{p(y_t|x_t; \theta) p(x_t|y^{t-1}; \theta)}{\int p(y_t|x_t; \theta) p(x_t|y^{t-1}; \theta) \, dx_t}
\] (1.2)

However, most macroeconomic researchers have interest in \( p(\theta|y^T) \) and \( p(y^T) \) rather than \( p(x_t|y^t; \theta) \).

In particular, the denominator in Eq (2), called normalizing constant or marginal data density, can be used for model assessment such as posterior odd test. In linear-Gaussian case it is possible to obtain it with Kalman filter. Otherwise, it could be approximated with particle filter.

2 Typical Estimating Method for DSGE Model

In macroeconomics, researchers usually use log-linearized Gaussian model of the economy. Accordingly, they calculate the likelihood for the observations through the Kalman Filter and then estimate the parameters based on Metropolis-Hasting (MH) Algorithm.\(^3\) Since this typical method

---

\(^2\)A variable with superscript \( t \) denotes a sequence of the variables from initial value to those at time \( t \), for instance, \( x^t = (x_0, x_1, ..., x_t) \), and \( y^t = (y_1, y_2, ..., y_t) \).

\(^3\)see Guerrón-Quintana and Nason (2012) for details
is well known, I briefly describe the process of the Kalman Filter and MH algorithm in below. In doing this, the likelihood can be obtained as a by-product of the Kalman Filter.

2.1 Kalman Filter

Given $x_0 \sim N(m_0, C_0)$,

- Predictive Step for state variable:
  $$x_t|x_{t-1}, \theta \sim N(a_t, R_t) \text{ where } a_t = G_t m_{t-1}, \quad R_t = G_t C_{t-1} G_t^T + \Sigma_w$$

- Predictive Step for observation:
  $$y_t|x_t, \theta \sim N(f_t, Q_t) \quad \text{ where } f_t = F_t' a_t, \quad Q_t = F_t' R_t F_t + \Sigma_\nu$$

- Filtering Step:
  $$x_t|y_t, \theta \sim N(m_t, C_t) \quad \text{ where } m_t = a_t + A_t(y_t - a_t) \quad C_t = R_t - A_t Q_t A_t' \quad A_t \equiv R_t F_t Q_t^{-1}$$

likelihood with Kalman Filter

$$\ln(\mathcal{L}(\theta|y^T)) \propto -\frac{1}{2} \sum_{t=1}^T \left( \ln(|Q_t|) + (y_t - f_t)' Q_t^{-1} (y_t - f_t) \right) \quad (2.1)$$

Once they have Kalman filter, they use Random Walk Metropolis-Hasting algorithm for the estimation of the parameters. The likelihood obtained from Kalman filter is used to choose desirable draws.

2.2 Algorithm 1: Random Walk Metropolis-Hasting

For $i = 1, \cdots , M$

1. Initialization: set $i = 0$ and initial $\theta^{(0)}$. Solve the model for $\theta^{(0)}$ and calculate $f(\cdot | \theta^{(0)})$ and $g(\cdot | \theta^{(0)})$ with Eq. (1). Then evaluate $p(\theta^{(0)})$ and $\mathcal{L}(\theta^{(0)}|y^T)$ from Eq. (3). Set $i = i + 1$.

2. Proposal draw: get a proposal draw $\theta^{(i*)} = \theta^{(i-1)} + \epsilon_i$, where $\epsilon_i \sim N(0, \Sigma_\epsilon)$.

- $\Sigma_\epsilon$ is a scaling matrix; when acceptance rate is too high we can scale up it by multiplying constant.
• A recommended optimal acceptance rate is around 23%

3. Solving the model: solve the model for \( \theta^{(i^*)} \) and calculate \( f(\cdot | \theta^{(i)}) \) and \( g(\cdot | \theta^{(i)}) \) with Eq. (1)
   building new state space representation.

4. Evaluating the proposal: Evaluate \( p(\theta^{(i^*)}) \) and \( \mathcal{L}(y^T | \theta^{(i^*)}) \) from Eq. (3)

5. Accept/Reject: Draw \( \rho_i \sim U(0, 1) \). If \( \rho_i \leq \frac{\mathcal{L}(\theta^{(i^*)} | y^T)p(\theta^{(i^*)})}{\mathcal{L}(\theta^{(i-1)} | y^T)p(\theta^{(i-1)})} \) set \( \theta^{(i)} = \theta^{(i^*)} \) otherwise \( \theta^{(i)} = \theta^{(i-1)} \). If \( i \leq M \) set \( i = i + 1 \) and go to 2. Otherwise stop.

Once I get the draws, I can approximate expected value of a function of the parameters, \( h(\theta) \), by \( \frac{1}{M} \sum_{i=1}^{M} h(\theta^{(i)}) \).

3 Particle Filters

In this section, I briefly review the four particle filters. First two filters are for state filtering and the others are for parameter learning. I assume all parameters are known for state filtering and omit it from most of the equations until I deal with parameter learning.

The idea behind of particle filter is the importance sampling. In order to approximate the target distribution \( p(x^t | y^t) \), one samples from proposal distribution \( q_{0:t}(x^t | y^t) \) with importance weight \( w_t = \frac{p(x^t | y^t)}{q_{0:t}(x^t | y^t)} \). Define \( q_{0:t}(x^t | y^t) \equiv q_t(x_t | x^{t-1}, y^t) q_{0:t-1}(x^{t-1} | y^{t-1}) \). We can compute importance weight recursively as following way:

\[
w_t = \frac{p(y_t | x_t) p(x_t | x_{t-1}) p(x_{t-1} | y^{t-1})}{p(y_t | y^{t-1}) q_t(x_t | x^{t-1}, y^t) q_{0:t-1}(x^{t-1} | y^{t-1})} \propto w_{t-1} \frac{p(y_t | x_t) p(x_t | x_{t-1})}{q_t(x_t | x^{t-1}, y^t)}
\]

(3.1)

The ratio of the densities in second part of Eq. (4) is called incremental importance weight. Given the draws \( \{x^{t,(i)}, w^{(i)}_t\}_{i=1}^{N} \), we can approximate expectations of a function \( f(x^t) \) of state variables as:

\[
E_q[f(x^t)] = \int f(x^t) \frac{p(x^t | y^t)}{q_{0:n}(x^t | y^t)} q_{0:t}(x^t | y^t) \, dx^t
\]

\[
E_q[f(x^t)] \approx \sum_{i=1}^{N} f(x^{t,(i)}) \hat{w}^{(i)}_t \quad \text{where} \quad \hat{w}^{(i)}_t = \frac{w^{(i)}_t}{\sum_{i=1}^{N} w^{(i)}_t}
\]

(3.2)

However, particle filtering has the weight degeneracy problem that only one particle is left as number of iterations increases. In order to mitigate this problem, “Resampling” step can be added.
end of an iteration in which effective number of particle is less than threshold. The effective sample size can be calculated by $\text{ESS} = \frac{1}{\sum_{i=1}^{N} (\hat{w}_i^t)^2}$. Thus, given known parameters, state filtering can be done following the method described above. Accordingly, when it comes to state filtering, choosing proposal density is crucial. I introduce Bootstrap Filter and Auxiliary Particle filter which adopt different proposal densities. On the other hand, in case with unknown parameters, things become little bit complicated. This is because simply including parameters into particle set would not successful. Applying a standard SMC algorithm to the Markov process of $\{x_t, \theta_t\}$ means that the parameter space would only be explored at the initialization of the algorithm. As a result of the successive resampling steps, after a certain time $t$, the approximation to $p(\theta|y^t)$ will only contain a single unique value for $\theta^4$. There have been two important attempts for this difficulty: artificial dynamics based approach (Liu and West (2001)) and sufficient statistics based approach (Storvik (2002)) as explained below.

3.1 Particle Filters with known parameters

**Bootstrap filter** Transition density is used as the proposal density in bootstrap filter, that is, $q_t(x_t|x_{t-1}, y_t) = p(x_t|x_{t-1})$, then incremental importance weight becomes the likelihood, $p(y_t|x_t)$, and the weight evolves $w_t \propto w_{t-1} p(y_t|x_t)$. As I mentioned in Introduction, however, this method does not take into account current observation, $y_t$, when drawing a sample from proposal distribution. We can use conditional transition density, $p(x_t|y_t, x_{t-1})$ to supplement bootstrap filter, but it is available only if the measurement equations are linear and Gaussian. If it is case, $w_t \propto w_{t-1} p(y_t|x_{t-1})$, that is, important weight independent from current particles. Accordingly, resample step can be done before sampling new particles: propagate-resample step can be reversed to resample-propagate step. The particles would be improved because now the proposal distribution reflects the information of current observation. Auxiliary particle filters are arising from this property.

**Auxiliary Particle Filter** Pitt and Shephard (1999) introduced proposal density $q_t(x_t, k|y^t)$ where $k$ is an auxiliary variable that indexes the particles in existence from time $t - 1$. The purpose of this auxiliary variable is to find a way to use the information in the current observation, $y_t$,  

\footnote{Kantas et al.}
to find the “good” particles within the existing set \( \left\{ x_{t-1}^{(i)} \right\}_{i=1}^{N} \) in order to form a better proposal distribution. Notice, from Eq. (2) and Eq. (4), 
\[
p(x_{t}|y^{t}) \approx \sum_{i=1}^{N} p(y_{t}|x_{t}^{(i)}) p(x_{t}^{(i)}|x_{t-1}^{(i)}) w_{t-1}^{(i)}.
\]
Pitt and Shephard (1999) idea is newly introduce target and proposal distribution as follows:
\[
p(x_{t}, k|y^{t}) \approx p(y_{t}|x_{t}) p(x_{t}|x_{t-1}^{(k)}) w_{t-1}^{(k)}
\]
\[
q(x_{t}, k|y^{t}) = p(y_{t}|g(x_{t-1}^{(k)})) p(x_{t}|x_{t-1}^{(k)}) w_{t-1}^{(k)}
\]
where \( g(x_{t-1}) = E(x_{t} | x_{t-1}) \) usually, and then the importance weight is expressed as
\[
w_{t} \propto \frac{p(y_{t}|x_{t})}{p(y_{t}|g(x_{t-1}))} \tag{3.3}
\]
Doucet and Johansen (2009), however, show that the performance of the APF will depend upon the signal to noise ratio in the state space model. When the ratio is low, it can mislead the set of particles away from interesting areas of support. Followings are the algorithm for auxiliary particle filter presented by Carvalho et al. (2010).

**Algorithm 2: Auxiliary Particle Filter (APF)**

1. Resample: \( \left\{ \tilde{x}_{t-1}^{(i)} \right\}_{i=1}^{N} \) from \( \left\{ x_{t-1}^{(i)} \right\}_{i=1}^{N} \) with weights \( w_{t}^{(i)} \propto p(y_{t}|g(x_{t-1}^{(i)})) \)
2. Propagate \( \tilde{x}_{t-1}^{(i)} \rightarrow \tilde{x}_{t}^{(i)} \): \( \left\{ \tilde{x}_{t}^{(i)} \right\}_{i=1}^{N} \sim p(x_{t} | \tilde{x}_{t-1}^{(i)}) \)
3. Resample: \( \left\{ x_{t}^{(i)} \right\}_{i=1}^{N} \) from \( \left\{ \tilde{x}_{t}^{(i)} \right\}_{i=1}^{N} \) with weights
\[
w_{t}^{(i)} \propto \frac{p(y_{t+1}|\tilde{x}_{t}^{(i)})}{p(y_{t+1}|g(\tilde{x}_{t}^{(i)}))}
\]

**Log-likelihood from APF** Notice that
\[
\ln p(y_{1}, ..., y_{T}|\theta) = \sum_{n=1}^{T} \ln p(y_{t}|y^{t-1}, \theta)
\]
Once we have done with sampling from the proposal according to the above algorithm, we can approximate the marginal distribution by:

\[ \hat{p}(y_t | y_{t-1}, \theta) = \frac{1}{N} \sum_{i=1}^{N} \frac{p(x_t^{(i)} | y_t)}{q_n(x_t^{(i)} | y_t)} = \frac{1}{N} \sum_{i=1}^{N} w_t^{(i)} \]

then,

\[ \ln L(\theta | y^T) \approx \sum_{t=1}^{T} \ln \hat{p}(y_t | y_{t-1}, \theta) = \sum_{t=1}^{T} \ln \left[ \frac{1}{N} \sum_{i=1}^{N} w_t^{(i)} \right] \]

Creal (2009), however, also mention that for a case in which resampling is performed not in every period, log-likelihood can be obtained as:

\[ \hat{p}(y_t | y_{t-1}, \theta) \approx \sum_{i=1}^{N} \hat{w}_{t-1}^{(i)} w_t^{(i)} \]

Noticing that \( \hat{w}_t^{(i)} \) is normalized weight,

\[ \ln L(\theta | y^T) \approx \sum_{t=1}^{T} \ln \hat{p}(y_t | y_{t-1}, \theta) = \sum_{t=1}^{T} \ln \left[ \sum_{i=1}^{N} \hat{w}_{t-1}^{(i)} w_t^{(i)} \right] \]

### 3.2 Particle Filters with unknown parameter

So far, the particle filtering is used for the state filtering and likelihood for observation under the assumption that all parameters are known. In this subsection, I relaxed this assumption and briefly introduce two important filters by which one can estimate the parameters as well as approximate state distribution.

**Liu and West (2001) Filter** They assume that the posterior distribution of the parameters can be approximated by mixture distribution:

\[ p(\theta | y^t) \approx \sum_{i=1}^{N} N \left( m^{(i)}; h^2 V_t \right) \]

where \( m^{(i)} = a \theta_t^{(i)} + (1 - a) \bar{\theta}, \ \theta_t = \frac{1}{N} \sum_{i=1}^{N} \theta_t^{(i)} \), and \( V_t = \frac{1}{N} \sum_{i=1}^{N} \left( \theta_t^{(i)} - \bar{\theta} \right) \left( \theta_t^{(i)} - \bar{\theta} \right)' \). A tuning parameter “\( a \)” determines the shrinkage and smoothness of the normal approximation and the desired value for “\( a \)” is suggested higher than 0.98 but less than 1. LW filtering algorithm
presented by Carvalho et al. (2010) is following:

1. Resample: \( \left\{ (\tilde{x}_t, \tilde{\theta}_t)^{(i)} \right\}_{i=1}^N \) from \( \left\{ (x_t, \theta_t)^{(i)} \right\}_{i=1}^N \) with weights
   \[ w_{t+1}^{(i)} \propto p \left( y_{t+1} \mid g \left( x_t^{(i)} \right), m_t^{(i)} \right) \]

2. Propagate \( \tilde{\theta}_t^{(i)} \rightarrow \tilde{\theta}_{t+1}^{(i)} \) : \( \left\{ \tilde{\theta}_{t+1}^{(i)} \right\}_{i=1}^N \sim p \left( \tilde{n}_t^{(i)}, V \right) \)

3. Propagate \( \tilde{x}_t^{(i)} \rightarrow \tilde{x}_{t+1}^{(i)} \) : \( \left\{ \tilde{x}_{t+1}^{(i)} \right\}_{i=1}^N \sim p \left( x_{t+1} \mid \tilde{x}_t^{(i)}, \tilde{\theta}_{t+1}^{(i)} \right) \)

4. Resample: \( \left\{ (x_{t+1}, \theta_{t+1})^{(i)} \right\}_{i=1}^N \) from \( \left\{ (\tilde{x}_{t+1}, \tilde{\theta}_{t+1})^{(i)} \right\}_{i=1}^N \) with weights
   \[ w_{t+1}^{(i)} \propto \frac{p \left( y_{t+1} \mid \tilde{x}_{t+1}^{(i)}, \tilde{\theta}_{t+1}^{(i)} \right)}{p \left( y_{t+1} \mid g \left( x_t^{(i)} \right), m_t^{(i)} \right)} \]

As we have seen above, the importance weight can be obtained before sampling state variables. Thus it makes resample-propagate framework as in APF.

**Storvik (2002) Filter** He assume that the posterior distribution \( p \left( \theta \mid x^t, y^t \right) \) depends on a low dimensional set of sufficient statistics, \( s_t \), that can be recursively updated: \( s_t = S(s_{t-1}, x_t, y_t) \).

As in below algorithm, Storvik filter can be considered as a Bootstrap filter with some additional steps because resampling is conducted after propagating. In other words, it uses blind proposal in that it does not take into account current observation, \( y_t \). The algorithm suggested by Storvik (2002) is following:

1. Propagate \( \left( x_t^{(i)} \rightarrow \tilde{x}_{t+1}^{(i)} \right) : \left\{ \tilde{x}_{t+1}^{(i)} \right\}_{i=1}^N \sim q \left( x_t^{(i)}, \theta^{(i)}, y_t+1 \right) \)

2. Resample: \( \left\{ (x_{t+1}, s_t)^{(i)} \right\}_{i=1}^N \) from \( \left\{ (\tilde{x}_{t+1}, s_t)^{(i)} \right\}_{i=1}^N \) with weights
   \[ w_{t+1}^{(i)} = \frac{p \left( y_{t+1} \mid \tilde{x}_{t+1}^{(i)}, \theta \right) p \left( \tilde{x}_{t+1}^{(i)} \mid x_t^{(i)}, \theta \right)}{q \left( x_t^{(i)}, \theta, y_{t+1} \right)} \]

3. Propagate \( \left( s_t^{(i)} \rightarrow \tilde{s}_{t+1}^{(i)} \right) : \tilde{s}_{t+1}^{(i)} = S(s_t^{(i)}, x_{t+1}^{(i)}, y_{t+1}) \).
4. Sample: $\theta^{(i)} \sim p\left(\theta|s_{t+1}^{(i)}\right)$

4 Particle Learning: PL

Carvalho et al. (2010) introduce particle learning method which is taking both of sufficient statistics of state variables and resample-propagate framework at the same time. PL directly samples from the particle approximation to the joint posterior distribution of states and conditional sufficient statistics for fixed parameters in a fully-adapt resample-propagate framework. Carvalho et al. (2010) shows that PL outperforms Liu and West (2001) in regards to accuracy and argues PL is comparable to MCMC samplers. Let $s_t$ and $s_{x_t}$ denote the parameter and state sufficient statistics satisfying deterministic updating rules $s_t = S(s_{t-1}, x_t, y_t)$ as in Storvik (2002) and $s_{x_t}^x = K(s_{x_{t-1}}^x, \theta, y_t)$ where $K$ is the Kalman filter recursions.

Algorithm 3: PL

1. Resample $\left(\tilde{\theta}, \tilde{s}_{t-1}^x, \tilde{s}_{t-1}\right)$ from $(\theta, s_{t-1}^x, s_{t-1})$ with weights $w_t \propto p(y_t|s_{t-1}^x, \theta)$
2. Sample $x_t$ from $p\left(x_t|\tilde{s}_{t-1}^x, \tilde{\theta}, y^t\right)$.
3. Update parameter sufficient statistics: $s_t = S(\tilde{s}_{t-1}, x_t, y_t)$
4. Sample $\theta$ from $p(\theta|s_t)$.
5. Update state sufficient statistics: $s_{x_t}^x = K(\tilde{s}_{x_{t-1}}^x, \theta, y_t)$.

5 Application to the simple model

In order to compare SMC method to MCMC, I try to replicate Johannes and Polson’s Example using both MCMC and PL. The model is called AR(1) with noise and is given by:

$$x_t = \alpha + \beta x_{t-1} + \varepsilon_t \quad \varepsilon_t \sim N(0, \tau^2)$$
$$y_t = x_t + \nu_t \quad \nu_t \sim N(0, \sigma^2)$$

I set the true parameters $\theta = (\alpha \ \beta \ \sigma^2 \ \tau^2) = (0 \ 0.9 \ 1 \ 0.5)$. I then generate $T = 200$ observations, $y_t$ for $t = 1, ..., T$ using Eq. (7). With known parameters and $x_0 = 0$, I compute log-likelihood from
APF with 10000 particles as well as Kalman filter which is presented in Table 1. I also calculate the likelihood with 1000 and 100000 particles. It improves the likelihood only in 1 decimal place as number of particles increases but it creates huge difference in running time. In addition, I compare those two filtering methods in term of state filtering. Figure 1 shows predicted state variable for each filtering method with the true values. The grey lines in upper panel show 5% and 95% quantiles of normal distribution with predictive mean, $m_t$, and variance, $C_t$, in Kalman Filter. The grey lines in lower panel show 5% and 95% quantiles of the particles in APF. It confirms that APF can compete with Kalman Filter.

I then assume prior distribution for $\theta = (\alpha, \beta, \sigma^2, \tau^2)$ to be $p(\theta) = p(\sigma^2)p(\tau^2)p(\alpha, \beta|\tau^2)$ where $\sigma^2 \sim IG(\psi_1, \Psi_1)$, $\tau^2 \sim IG(\psi_2, \Psi_2)$ and $\alpha, \beta|\tau^2 \sim MN(\psi_3, \tau^2\Psi_3)$. The weight used in initial resampling step is the one step ahead predictive likelihood $p(y_{t+1}|x_t, \theta) \sim N(\alpha + \beta x_t, \sigma^2 + \tau^2)$ which is given by

$$w\left( (x_t, \theta)^{(i)} \right) \propto \frac{1}{\sqrt{(\sigma^2)^{(i)} + (\tau^2)^{(i)}}} \exp\left( \frac{1}{2} \frac{(y_{t+1} - \alpha^{(i)} - \beta^{(i)}x_t^{(i)})^2}{(\sigma^2)^{(i)} + (\tau^2)^{(i)}} \right)^{(5.2)}$$

The updated state distribution is $p(x_t|x_{t-1}, \theta, y_t) \propto p(y_t|x_t, \theta)p(x_t|x_{t-1}, \theta) \sim N(\mu_t, \Omega_t^2)$ where

$$\mu_t = \frac{y_t}{\sigma^2 + \tau^2} + \frac{\alpha + \beta x_{t-1}}{\tau^2}$$
$$\Omega_t^2 = \frac{1}{\sigma^2 + \tau^2}$$

Given sufficient statistics, $s_t$, the posterior distribution for parameters is now $p(\theta|s_{t-1}) = p(\alpha, \beta|\tau^2, s_{t-1})p(\sigma^2|s_{t-1})p(\tau^2|s_{t-1})$.

$$p(\sigma^2|s_t) \sim IG\left( \frac{\psi_{1,t}}{2}, \frac{\Psi_{1,t}}{2} \right), \quad p(\tau^2|s_t) \sim IG\left( \frac{\psi_{2,t}}{2}, \frac{\Psi_{2,t}}{2} \right)$$
$$p(\alpha, \beta|\tau^2, s_t) \sim N_2\left( \frac{\psi_{3,t}}{\tau^2}, \frac{1}{\tau^2\Psi_{3,t}} \right)$$

(5.3)

Table 1: Log-likelihood

<table>
<thead>
<tr>
<th>Log-likelihood</th>
<th>Kalman Filter</th>
<th>Auxiliary Particle Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>-342.91639713437</td>
<td>-346.170709961134</td>
<td></td>
</tr>
</tbody>
</table>
Let $Z_t = [1 \ x_t]$ and the vector of sufficient statistics $s_t = (\Psi_{1,t} \ \Psi_{2,t} \ \psi_{3,t} \ \Psi_{3,t})$ then, corresponding transition kernel $S(s_{t-1}, x_t, y_t)$ is given by:

\[
\begin{align*}
\Psi_{1,t} &= (y_t - x_t)^2 + \Psi_{1,t-1} \\
\Psi_{2,t} &= \Psi_{2,t-1} + \psi_{3,t-1}' \Psi_{3,t-1} \psi_{3,t-1} + x_t' x_t - \psi_{3,t}' \Psi_{3,t} \psi_{3,t} \\
\psi_{3,t} &= \Psi_{3,t}^{-1} (\Psi_{3,t-1} \psi_{3,t-1} + Z_{t-1} x_t) \\
\Psi_{3,t} &= \Psi_{3,t-1} + Z_t Z_t'
\end{align*}
\] (5.4)

Figure 1: Simulated State Variables
The hyperparameters are deterministic and given by \( \psi_{1,t} = \psi_{1,t-1} + 1 \) and \( \psi_{2,t} = \psi_{2,t-1} + 1 \). For the parameter estimation, I additionally set the initial prior parameters 
\[ \Theta_0 = (\psi_{1,0}, \psi_{1,0}, \psi_{2,0}, \psi_{3,0}, \psi_{3,0}) = (5, 5, 5, 2.5, 0, 0.9, I_2). \] Accordingly, the algorithm for particle learning can be written as:

1. Resample \((\tilde{\theta}, \tilde{s}_{t-1}, \tilde{s}_t)\) from \((\theta, s_{t-1}, s_t)\) with weights \( w_t \propto \mathcal{N}(\alpha + \beta x_{t-1}, \sigma^2 + \tau^2) \)

2. Sample \( x_t \) from \( \mathcal{N}\left(\Omega^2 \left(\frac{y_t}{\sigma^2} + \frac{\alpha + \beta x_{t-1}}{\tau^2}\right), \Omega^2\right) \)

3. Update parameter sufficient statistics: \( s_t = S(\tilde{s}_{t-1}, x_t, y_t) \)

4. Sample \( \theta \) from \( p(\theta|s_t) \sim \text{Eq.}(9) \).

Figure 2. shows the estimated parameters based on PL which is getting closer to the true value of the parameter as it is updated; Red dashed lines indicates the true parameter value, black lines represent median of the estimated parameter value and the grey lines are corresponding 0.025 and 0.975 quantiles. For the sake of comparison, I also estimate the parameters with Random Walk MH algorithm rather than Gibbs sampler even though, in my example, the distributions can be obtained analytically. This is because I intend to compare the SMC method with a method typically used in DSGE model estimation which is explained above. I run MH algorithm 100,000 times and burn first 50,000 iterations. I draw each of the parameters \( \alpha, \beta, \sigma, \) and \( \tau \) from the proposal distribution which is univariate normal distribution centered at previous accepted parameter values. I chose jumping parameter \( c = 0.083 \), then the acceptance rate become 24.33\%. In order to get the parameters \( \sigma^2 \) and \( \tau^2 \), I squared the accepted draws for \( \sigma \) and \( \tau \). I summarize posterior quantiles obtained by PL as well as MCMC in Table 2. For all the parameters, true parameter values are within 95% credible intervals. For all parameters but \( \tau^2 \), the median of PL closer to true value than those of Random walk MH algorithm to the true values.

6 Conclusion

In this paper I have summarized particle filters as an alternative to MCMC method. Depending on the order of sampling and resampling, the filters can be categorized in two. One is propagat-
Figure 2: Median and 95% credible intervals of the parameters

desirable in that the information of current observation is used to sample draws. In addition, I have learned PL suggested by Carvalho et al. (2010) for parameter estimation. The purpose of present paper is to see whether PL outperforms MCMC as they argue in their paper. To do this, I construct simple linear Gaussian state space model called AR(1) with noise by Johannes and Polson. Through Kalman filter and APF, I obtained the likelihood for the observations and predicted stated variables. According to the likelihood, Kalman filter is still better than APF for linear case, but it is comparable. I then estimate the parameters using random walk MH algorithm and PL. Based on posterior quantiles, PL gives better
Table 2: Comparison of quantiles between PL and MCMC

<table>
<thead>
<tr>
<th>Parameters (True Value)</th>
<th>MCMC</th>
<th>Particle Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5%</td>
<td>Median</td>
</tr>
<tr>
<td>α (0.0)</td>
<td>-0.191</td>
<td>-0.063</td>
</tr>
<tr>
<td>β (0.9)</td>
<td>0.700</td>
<td>0.819</td>
</tr>
<tr>
<td>σ² (1.0)</td>
<td>0.241</td>
<td>0.585</td>
</tr>
<tr>
<td>τ² (0.5)</td>
<td>0.166</td>
<td>0.481</td>
</tr>
</tbody>
</table>

approximation for the true parameters and it is also better in running time.
References


Michael Johannes and Nick Polson. Particle filtering and parameter learning.

Nicholas Kantas, Arnaud Doucet, Sumeetpal Sindhu Singh, and Jan Marian Maciejowski. An overview of sequential monte carlo methods for parameter estimation in general state-space models.


Geir Storvik. Particle filters for state-space models with the presence of unknown static parameters.

Appendix I: Results from the MCMC

Figure 3: MCMC: Posterior Distribution of the Parameters

Figure 3. represents the posterior distribution of the parameters obtained by MCMC. Red vertical lines indicate true parameter values. Mode of the draws are not coincident with the true values but as in Table 2, $\alpha$ and $\tau$ are well captured relative to others.
I plot Figure 4. to see the convergency of the algorithm. As it shows, cumulative mean of the parameters are converges to some point. However, it seems to be far from its true values except for $\tau^2$. 

Figure 4: MCMC: Cumulative mean of the parameters
Figure 5: MCMC: Draws of the parameters