Particle Filtering

Frank Schorfheide
University of Pennsylvania
Gerzensee Ph.D. Course on Bayesian Macroeconometrics

May 28, 2019
• Linear DSGE model leads to

\[
\begin{align*}
y_t &= \Psi_0(\theta) + \Psi_1(\theta)t + \Psi_2(\theta)s_t + u_t, \quad u_t \sim N(0, \Sigma_u), \\
s_t &= \Phi_1(\theta)s_{t-1} + \Phi_\epsilon(\theta)\epsilon_t, \quad \epsilon_t \sim N(0, \Sigma_\epsilon).
\end{align*}
\]

• Nonlinear DSGE model leads to

\[
\begin{align*}
y_t &= \Psi(s_t, t; \theta) + u_t, \quad u_t \sim F_u(\cdot; \theta) \\
s_t &= \Phi(s_{t-1}, \epsilon_t; \theta), \quad \epsilon_t \sim F_\epsilon(\cdot; \theta).
\end{align*}
\]
• While DSGE models are inherently nonlinear, the nonlinearities are often small and decision rules are approximately linear.

• One can add certain features that generate more pronounced nonlinearities:
  • stochastic volatility;
  • markov switching coefficients;
  • asymmetric adjustment costs;
  • occasionally binding constraints.
• There are many particle filters...

• We will focus on three types:
  • Bootstrap PF
  • A generic PF
  • A conditionally-optimal PF
Filtering - General Idea

- State-space representation of linearized DSGE model
  \[ y_t = \Psi_0(\theta) + \Psi_1(\theta)t + \Psi_2(\theta)s_t + u_t \]
  \[ s_t = \Phi_1(\theta)s_t + \Phi_\epsilon(\theta)\epsilon_t \]
- Likelihood function:
  \[ p(Y_{1:T}|\theta) = \prod_{t=1}^{T} p(y_t|Y_{1:t-1}, \theta) \]

- A filter generates a sequence of conditional distributions \( s_t | Y_{1:t} \).
- Iterations:
  - Initialization at time \( t - 1 \): \( p(s_{t-1}|Y_{1:t-1}, \theta) \)
  - Forecasting \( t \) given \( t - 1 \):
    1. Transition equation: \( p(s_t|Y_{1:t-1}, \theta) = \int p(s_t|s_{t-1}, Y_{1:t-1}, \theta)p(s_{t-1}|Y_{1:t-1}, \theta)ds_{t-1} \)
    2. Measurement equation: \( p(y_t|Y_{1:t-1}, \theta) = \int p(y_t|s_t, Y_{1:t-1}, \theta)p(s_t|Y_{1:t-1}, \theta)ds_t \)
  - Updating with Bayes theorem. Once \( y_t \) becomes available:
  \[ p(s_t|Y_{1:t}, \theta) = p(s_t|y_t, Y_{1:t-1}, \theta) = \frac{p(y_t|s_t, Y_{1:t-1}, \theta)p(s_t|Y_{1:t-1}, \theta)}{p(y_t|Y_{1:t-1}, \theta)} \]
Bootstrap Particle Filter – Idea

Time = t

1. PROPAGATION

\( \tilde{x}_{t+1}^{(i)} \sim f(\cdot|x_t^{(i)}) \)

\( \pi_{t+1|0:t}(x_{t+1}|y_{0:t}) \)

\( \{x_t^{(i)}\} \)

2. WEIGHTING

\( \omega_{t+1}^{(i)} \propto g(y_{t+1}|\tilde{x}_{t+1}^{(i)}) \)

3. RESAMPLING

\( \pi_{t+1|0:t+1}(x_{t+1}|y_{0:t+1}) \)

\( \{\tilde{x}_{t+1}^{(i)} = x_t^{(i)}\} \)

\( \{x_t^{(j)}\} \)

Time = t+1
Bootstrap Particle Filter

1 Initialization. Draw the initial particles from the distribution \( s^j_0 \sim p(s_0) \) and set \( W^j_0 = 1, j = 1, \ldots, M \).

2 Recursion. For \( t = 1, \ldots, T \):
   1 Forecasting \( s_t \). Propagate the period \( t-1 \) particles \( \{s^{j}_{t-1}, W^j_{t-1}\} \) by iterating the state-transition equation forward:

\[
\tilde{s}^j_t = \Phi(s^j_{t-1}, \epsilon^j_t; \theta), \quad \epsilon^j_t \sim F_{\epsilon}(\cdot; \theta). \tag{1}
\]

An approximation of \( \mathbb{E}[h(s_t)|Y_{1:t-1}, \theta] \) is given by

\[
\hat{h}_{t,M} = \frac{1}{M} \sum_{j=1}^{M} h(\tilde{s}^j_t)W^j_{t-1}. \tag{2}
\]
1 Initialization.

2 Recursion. For $t = 1, \ldots, T$:

1. Forecasting $s_t$.
2. Forecasting $y_t$. Define the incremental weights

$$\tilde{w}_t^j = p(y_t | \tilde{s}_t^j, \theta).$$

The predictive density $p(y_t | Y_{1:t-1}, \theta)$ can be approximated by

$$\hat{p}(y_t | Y_{1:t-1}, \theta) = \frac{1}{M} \sum_{j=1}^{M} \tilde{w}_t^j W_{t-1}^j.$$ (4)

If the measurement errors are $N(0, \Sigma_u)$ then the incremental weights take the form

$$\tilde{w}_t^j = (2\pi)^{-n/2} |\Sigma_u|^{-1/2} \exp \left\{ -\frac{1}{2} (y_t - \Psi(\tilde{s}_t^j, t; \theta))' \Sigma_u^{-1} (y_t - \Psi(\tilde{s}_t^j, t; \theta)) \right\},$$ (5)

where $n$ here denotes the dimension of $y_t$. 
Bootstrap Particle Filter

1 **Initialization.**

2 **Recursion.** For $t = 1, \ldots, T$:
   1 **Forecasting** $s_t$.
   2 **Forecasting** $y_t$. Define the incremental weights

   $$\tilde{w}_t^i = p(y_t | \tilde{s}_t^i, \theta).$$  \hspace{1cm} (6)

3 **Updating.** Define the normalized weights

   $$\tilde{W}_t^i = \frac{\tilde{w}_t^i W_t^i}{\frac{1}{M} \sum_{j=1}^M \tilde{w}_t^j W_t^j}. \hspace{1cm} (7)$$

   An approximation of $\mathbb{E}[h(s_t) | Y_{1:t}, \theta]$ is given by

   $$\tilde{h}_{t,M} = \frac{1}{M} \sum_{j=1}^M h(\tilde{s}_t^j) \tilde{W}_t^j. \hspace{1cm} (8)$$
Bootstrap Particle Filter

1 **Initialization.**
2 **Recursion.** For $t = 1, \ldots, T$:
   1 Forecasting $s_t$.
   2 Forecasting $y_t$.
3 **Updating.**
4 **Selection (Optional).** Resample the particles via multinomial resampling. Let $\{s_t^j\}_{j=1}^M$ denote $M$ iid draws from a multinomial distribution characterized by support points and weights $\{\tilde{s}_t^j, \tilde{W}_t^j\}$ and set $W_t^j = 1$ for $j = 1, \ldots, M$.
   An approximation of $E[h(s_t) | Y_{1:t}, \theta]$ is given by
   $$
   \bar{h}_{t,M} = \frac{1}{M} \sum_{j=1}^M h(s_t^j) W_t^j.
   $$

3 **Likelihood Approximation.** The approximation of the log likelihood function is given by
   $$
   \ln \hat{\rho}(Y_{1:T} | \theta) = \sum_{t=1}^T \ln \left( \frac{1}{M} \sum_{j=1}^M \tilde{\omega}_t^j W_{t-1}^j \right).
   $$

Frank Schorfheide
Particle Filtering
Asymptotics

• The convergence results can be established recursively, starting from the assumption

\[ \bar{h}_{t-1,M} \xrightarrow{a.s.} \mathbb{E}[h(s_{t-1})|Y_{1:t-1}], \]

\[ \sqrt{M}(\bar{h}_{t-1,M} - \mathbb{E}[h(s_{t-1})|Y_{1:t-1}]) \Rightarrow N(0, \Omega_{t-1}(h)). \]

• Forward iteration: draw \( s_t \) from \( g_t(s_t|s_{t-1}^j) = p(s_t|s_{t-1}^j). \)

• Decompose

\[ \hat{h}_{t,M} - \mathbb{E}[h(s_t)|Y_{1:t-1}] \]

\[ = \frac{1}{M} \sum_{j=1}^{M} \left( h(\tilde{s}_t^j) - \mathbb{E}_{p(\cdot|s_{t-1}^j)}[h] \right) W_{t-1}^j \]

\[ + \frac{1}{M} \sum_{j=1}^{M} \left( \mathbb{E}_{p(\cdot|s_{t-1}^j)}[h] W_{t-1}^j - \mathbb{E}[h(s_t)|Y_{1:t-1}] \right) \]

\[ = I + II, \]

• Both \( I \) and \( II \) converge to zero (and potentially satisfy CLT).
Asymptotics

- Updating step approximates

\[
\mathbb{E}[h(s_t) | Y_{1:t}] = \frac{\int h(s_t)p(y_t | s_t)p(s_t | Y_{1:t-1})ds_t}{\int p(y_t | s_t)p(s_t | Y_{1:t-1})ds_t} \approx \frac{1}{M} \sum_{j=1}^{M} h(\tilde{s}_t^j) \tilde{w}_t^j W_{t-1}^j
\]  

(12)

- Define the normalized incremental weights as

\[
v_t(s_t) = \frac{p(y_t | s_t)}{\int p(y_t | s_t)p(s_t | Y_{1:t-1})ds_t}.
\]  

(13)

- Under suitable regularity conditions, the Monte Carlo approximation satisfies a CLT of the form

\[
\sqrt{M}(\tilde{h}_t - \mathbb{E}[h(s_t) | Y_{1:t}]) \implies N(0, \tilde{\Omega}_t(h)) \quad \tilde{\Omega}_t(h) = \Omega_t(v_t(s_t)(h(s_t) - \mathbb{E}[h(s_t) | Y_{1:t}]))
\]  

(14)

- Distribution of particle weights matters for accuracy! \implies Resampling!
The Role of Measurement Errors

- Measurement errors may not be intrinsic to DSGE model.

- Bootstrap filter needs non-degenerate $p(y_t|s_t, \theta)$ for incremental weights to be well defined.

- Decreasing the measurement error variance $\Sigma_u$, holding everything else fixed, increases the variance of the particle weights, and reduces the accuracy of Monte Carlo approximation.
**Initialization.** Same as BS PF

**Recursion.** For $t = 1, \ldots, T$:

1. **Forecasting** $s_t$. Draw $\tilde{s}_t^j$ from density $g_t(\tilde{s}_t^j|s_{t-1}^j, \theta)$ and define

   $$
   \omega_t^j = \frac{p(\tilde{s}_t^j|s_{t-1}^j, \theta)}{g_t(\tilde{s}_t^j|s_{t-1}^j, \theta)}.
   $$

   An approximation of $\mathbb{E}[h(s_t)|Y_{1:t-1}, \theta]$ is given by

   $$
   \hat{h}_{t,M} = \frac{1}{M} \sum_{j=1}^{M} h(\tilde{s}_t^j) \omega_t^j W_{t-1}^j.
   $$

2. **Forecasting** $y_t$. Define the incremental weights

   $$
   \tilde{w}_t^j = p(y_t|\tilde{s}_t^j, \theta)\omega_t^j.
   $$

   The predictive density $p(y_t|Y_{1:t-1}, \theta)$ can be approximated by

   $$
   \hat{p}(y_t|Y_{1:t-1}, \theta) = \frac{1}{M} \sum_{j=1}^{M} \tilde{w}_t^j W_{t-1}^j.
   $$

**Updating.** Same as BS PF

**Selection.** Same as BS PF
• Conditionally-optimal importance distribution:

\[ g_t(\tilde{s}_t|s_{t-1}^j) = p(\tilde{s}_t|y_t, s_{t-1}^j). \]

This is the posterior of \( s_t \) given \( s_{t-1}^j \). Typically infeasible, but a good benchmark.

• Approximately conditionally-optimal distributions: from linearize version of DSGE model or approximate nonlinear filters.

• Conditionally-linear models: do Kalman filter updating on a subvector of \( s_t \). Example:

\[
\begin{align*}
y_t &= \Psi_0(m_t) + \Psi_1(m_t)t + \Psi_2(m_t)s_t + u_t, \quad u_t \sim N(0, \Sigma_u), \\
\Phi_0(m_t) + \Phi_1(m_t)s_{t-1} + \Phi_\epsilon(m_t)\epsilon_t, \quad \epsilon_t \sim N(0, \Sigma_\epsilon),
\end{align*}
\]

where \( m_t \) follows a discrete Markov-switching process.
More on Conditionally-Linear Models

- State-space representation is linear conditional on $m_t$.

- Write

$$p(m_t, s_t \mid Y_{1:t}) = p(m_t \mid Y_{1:t})p(s_t \mid m_t, Y_{1:t}),$$

where

$$s_t \mid (m_t, Y_{1:t}) \sim N(\bar{s}_t \mid t(m_t), P_t \mid t(m_t)).$$

- Vector of means $\bar{s}_t \mid t(m_t)$ and the covariance matrix $P_t \mid t(m_t)$ are sufficient statistics for the conditional distribution of $s_t$.

- Approximate $(m_t, s_t) \mid Y_{1:t}$ by $\{m^i_t, \bar{s}^i_t \mid t, P^i_t, W^i_t \}^{N}_{i=1}$.

- The swarm of particles approximates

$$\int h(m_t, s_t)p(m_t, s_t, Y_{1:t})d(m_t, s_t)$$

$$= \int \left[ \int h(m_t, s_t)p(s_t \mid m_t, Y_{1:t})ds_t \right]p(m_t \mid Y_{1:t})dm_t$$

$$\approx \frac{1}{M} \sum_{i=1}^{M} \left[ \int h(m^i_t, s^i_t)p_N(s_t \mid \bar{s}^i_t \mid t, P^i_t \mid t)ds_t \right] W^i_t.$$
More on Conditionally-Linear Models

• We used Rao-Blackwellization to reduce variance:

\[ \nabla[h(s_t, m_t)] = \mathbb{E}[\nabla[h(s_t, m_t)|m_t]] + \nabla[\mathbb{E}[h(s_t, m_t)|m_t]] \geq \nabla[\mathbb{E}[h(s_t, m_t)|m_t]] \]

• To forecast the states in period \( t \), generate \( \tilde{m}^i_t \) from \( g_t(\tilde{m}_t|\tilde{m}_{t-1}^j) \) and define:

\[ \omega^j_t = \frac{p(\tilde{m}_t^j|m_{t-1}^j)}{g_t(\tilde{m}_t^j|m_{t-1}^j)}. \quad (22) \]

• The Kalman filter forecasting step can be used to compute:

\[ \begin{align*}
\tilde{s}^i_{t|t-1} &= \Phi_0(\tilde{m}_t^j) + \Phi_1(\tilde{m}_t^j)s_{t-1}^i \\
\tilde{P}_t^i &= \Phi_c(\tilde{m}_t^j)\Sigma_c(\tilde{m}_t^j)\Phi_c(\tilde{m}_t^j)'
\end{align*} \]

\[ \begin{align*}
\tilde{y}^i_{t|t-1} &= \Psi_0(\tilde{m}_t^j) + \Psi_1(\tilde{m}_t^j)t + \Psi_2(\tilde{m}_t^j)\tilde{s}_{t|t-1}^i \\
F_t^i &= \Psi_2(\tilde{m}_t^j)\tilde{P}_{t|t-1}^i \Psi_2(\tilde{m}_t^j)' + \Sigma_u. \end{align*} \quad (23) \]
• Then,
\[
\int h(m_t, s_t)p(m_t, s_t|Y_{1:t-1})d(m_t, s_t)
\]
\begin{align*}
= & \int \left[ \int h(m_t, s_t)p(s_t|m_t, Y_{1:t-1})ds_t \right] p(m_t|Y_{1:t-1})dm_t \\
\approx & \frac{1}{M} \sum_{j=1}^{M} \left[ \int h(m^j_t, s^j_t)p_N(s^j_t|\tilde{s}^j_{t|t-1}, P^j_{t|t-1})ds_t \right] \omega_t^j \omega_t^j.
\end{align*}

The likelihood approximation is based on the incremental weights
\[
\tilde{\omega}_t^j = p_N(y_t|\tilde{y}^j_{t|t-1}, F^j_{t|t-1}) \omega_t^j.
\]

• Conditional on \( \tilde{m}_t^j \) we can use the Kalman filter once more to update the information about \( s_t \) in view of the current observation \( y_t \):
\[
\begin{align*}
\tilde{s}^j_{t|t} & = \tilde{s}^j_{t|t-1} + P^j_{t|t-1} \psi_2(\tilde{m}_t^j)'(F^j_{t|t-1})^{-1}(y_t - \tilde{y}^j_{t|t-1}) \\
\tilde{P}^j_{t|t} & = \tilde{P}^j_{t|t-1} - P^j_{t|t-1} \psi_2(\tilde{m}_t^j)'(F^j_{t|t-1})^{-1} \psi_2(\tilde{m}_t^j)P^j_{t|t-1}.
\end{align*}
\]
1 Initialization.

2 Recursion. For $t = 1, \ldots, T$:

1 Forecasting $s_t$. Draw $\tilde{m}_t^j$ from density $g_t(\tilde{m}_t^j | m_{t-1}^j, \theta)$, calculate the importance weights $\omega_t^j$ in (22), and compute $\tilde{s}_{t|t-1}^j$ and $P_{t|t-1}^j$ according to (23). An approximation of $\mathbb{E}[h(s_t, m_t) | Y_{1:t-1}, \theta]$ is given by (25).

2 Forecasting $y_t$. Compute the incremental weights $\tilde{w}_t^j$ according to (25). Approximate the predictive density $p(y_t | Y_{1:t-1}, \theta)$ by

$$\hat{p}(y_t | Y_{1:t-1}, \theta) = \frac{1}{M} \sum_{j=1}^M \tilde{w}_t^j W_{t-1}^j.$$ (27)

3 Updating. Define the normalized weights

$$\tilde{W}_t^j = \frac{\tilde{w}_t^j W_{t-1}^j}{\frac{1}{M} \sum_{j=1}^M \tilde{w}_t^j W_{t-1}^j}$$ (28)

and compute $\tilde{s}_{t|t}^j$ and $\tilde{P}_{t|t}^j$ according to (26). An approximation of $\mathbb{E}[h(m_t, s_t) | Y_{1:t}, \theta]$ can be obtained from $\{\tilde{m}_t^j, \tilde{s}_{t|t}^j, \tilde{P}_{t|t}^j, \tilde{W}_t^j\}$.

4 Selection.

3 Likelihood Approximation.
Nonlinear and Partially Deterministic State Transitions

- Example:
  \[ s_{1,t} = \Phi_1(s_{t-1}, \epsilon_t), \quad s_{2,t} = \Phi_2(s_{t-1}), \quad \epsilon_t \sim \mathcal{N}(0, 1). \]

- Generic filter requires evaluation of \( p(s_t|s_{t-1}) \).

- Define \( \varsigma_t = [s_t', \epsilon_t']' \) and add identity \( \epsilon_t = \epsilon_t \) to state transition.

- Factorize the density \( p(\varsigma_t|\varsigma_{t-1}) \) as
  \[ p(\varsigma_t|\varsigma_{t-1}) = p^e(\epsilon_t)p(s_{1,t}|s_{t-1}, \epsilon_t)p(s_{2,t}|s_{t-1}). \]
  where \( p(s_{1,t}|s_{t-1}, \epsilon_t) \) and \( p(s_{2,t}|s_{t-1}) \) are pointmasses.

- Sample innovation \( \epsilon_t \) from \( g^e_t(\epsilon_t|s_{t-1}) \).

- Then
  \[ \omega^j_t = \frac{p(\tilde{\varsigma}^j_t|\varsigma_{t-1})}{g^e_t(\tilde{\varsigma}^j_t|\varsigma_{t-1})} = \frac{p^e(\epsilon_t)p(s_{1,t}^j|s_{t-1}^j, \tilde{\epsilon}^j_t)p(s_{2,t}^j|s_{t-1}^j)}{g^e_t(\epsilon_t|s_{t-1}^j)p(s_{1,t}^j|s_{t-1}^j, \tilde{\epsilon}^j_t)p(s_{2,t}^j|s_{t-1}^j)} = \frac{p^e(\epsilon_t)}{g^e_t(\epsilon_t|s_{t-1}^j)}. \]
Degenerate Measurement Error Distributions

• Our discussion of the conditionally-optimal importance distribution suggests that in the absence of measurement errors, one has to solve the system of equations

\[ y_t = \Psi(\Phi(s_{t-1}^j, \tilde{\epsilon}_t^j)), \]

to determine \( \tilde{\epsilon}_t^j \) as a function of \( s_{t-1}^j \) and the current observation \( y_t \).

• Then define

\[ \omega_t^j = p^e(\tilde{\epsilon}_t^j) \quad \text{and} \quad s_t^j = \Phi(s_{t-1}^j, \tilde{\epsilon}_t^j). \]

• Difficulty: one has to find all solutions to a nonlinear system of equations.

• While resampling duplicates particles, the duplicated particles do not mutate, which can lead to a degeneracy.
Next Steps

• We will now apply PFs to linearized DSGE models.

• This allows us to compare the Monte Carlo approximation to the “truth.”

• Small-scale New Keynesian DSGE model

• Smets-Wouters model
### Parameter Values For Likelihood Evaluation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$\theta^m$</th>
<th>$\theta^l$</th>
<th>Parameter</th>
<th>$\theta^m$</th>
<th>$\theta^l$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau$</td>
<td>2.09</td>
<td>3.26</td>
<td>$\kappa$</td>
<td>0.98</td>
<td>0.89</td>
</tr>
<tr>
<td>$\psi_1$</td>
<td>2.25</td>
<td>1.88</td>
<td>$\psi_2$</td>
<td>0.65</td>
<td>0.53</td>
</tr>
<tr>
<td>$\rho_r$</td>
<td>0.81</td>
<td>0.76</td>
<td>$\rho_g$</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>$\rho_z$</td>
<td>0.93</td>
<td>0.89</td>
<td>$r^{(A)}$</td>
<td>0.34</td>
<td>0.19</td>
</tr>
<tr>
<td>$\pi^{(A)}$</td>
<td>3.16</td>
<td>3.29</td>
<td>$\gamma^{(Q)}$</td>
<td>0.51</td>
<td>0.73</td>
</tr>
<tr>
<td>$\sigma_r$</td>
<td>0.19</td>
<td>0.20</td>
<td>$\sigma_g$</td>
<td>0.65</td>
<td>0.58</td>
</tr>
<tr>
<td>$\sigma_z$</td>
<td>0.24</td>
<td>0.29</td>
<td>$\ln p(Y</td>
<td>\theta)$</td>
<td>-306.5</td>
</tr>
</tbody>
</table>
Likelihood Approximation

\[ \ln \hat{p}(y_t | Y_{1:t-1}, \theta^m) \quad \text{vs.} \quad \ln p(y_t | Y_{1:t-1}, \theta^m) \]

Notes: The results depicted in the figure are based on a single run of the bootstrap PF (dashed), the conditionally-optimal PF (dotted), and the Kalman filter (solid).
Notes: The results depicted in the figure are based on a single run of the bootstrap PF (dashed), the conditionally-optimal PF (dotted), and the Kalman filter (solid).
Notes: Density estimate of $\hat{\Delta}_1 = \ln \hat{p}(Y_{1:T}|\theta) - \ln p(Y_{1:T}|\theta)$ based on $N_{\text{run}} = 100$ runs of the PF. Solid line is $\theta = \theta^m$; dashed line is $\theta = \theta^l$ ($M = 40,000$).
Notes: Density estimate of $\hat{\Delta}_1 = \ln \hat{p}(Y_{1:T}|\theta) - \ln p(Y_{1:T}|\theta)$ based on $N_{\text{run}} = 100$ runs of the PF. Solid line is bootstrap particle filter ($M = 40,000$); dotted line is conditionally optimal particle filter ($M = 400$).
### Summary Statistics for Particle Filters

<table>
<thead>
<tr>
<th></th>
<th>Bootstrap</th>
<th>Cond. Opt.</th>
<th>Auxiliary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Particles $M$</td>
<td>40,000</td>
<td>400</td>
<td>40,000</td>
</tr>
<tr>
<td>Number of Repetitions</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
</tbody>
</table>

**High Posterior Density: $\theta = \theta^m$**

<table>
<thead>
<tr>
<th></th>
<th>$\hat{\Delta}_1$</th>
<th>$\hat{\Delta}_1$</th>
<th>$\hat{\Delta}_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias</td>
<td>-1.39</td>
<td>-0.10</td>
<td>-2.83</td>
</tr>
<tr>
<td>StdD</td>
<td>2.03</td>
<td>0.37</td>
<td>1.87</td>
</tr>
<tr>
<td>Bias</td>
<td>0.32</td>
<td>-0.03</td>
<td>-0.74</td>
</tr>
</tbody>
</table>

**Low Posterior Density: $\theta = \theta^l$**

<table>
<thead>
<tr>
<th></th>
<th>$\hat{\Delta}_1$</th>
<th>$\hat{\Delta}_1$</th>
<th>$\hat{\Delta}_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bias</td>
<td>-7.01</td>
<td>-0.11</td>
<td>-6.44</td>
</tr>
<tr>
<td>StdD</td>
<td>4.68</td>
<td>0.44</td>
<td>4.19</td>
</tr>
<tr>
<td>Bias</td>
<td>-0.70</td>
<td>-0.02</td>
<td>-0.50</td>
</tr>
</tbody>
</table>

**Notes:**

$\hat{\Delta}_1 = \ln \hat{p}(Y_{1:T}|\theta) - \ln p(Y_{1:T}|\theta)$ and $\hat{\Delta}_2 = \exp[\ln \hat{p}(Y_{1:T}|\theta) - \ln p(Y_{1:T}|\theta)] - 1$. Results are based on $N_{run} = 100$ runs of the particle filters.
Notes: Solid lines represent results from Kalman filter. Dashed lines correspond to bootstrap particle filter ($M = 40,000$) and dotted lines correspond to conditionally-optimal particle filter ($M = 400$). Results are based on $N_{run} = 100$ runs of the filters.
Notes: Solid lines represent results from Kalman filter. Dashed lines correspond to bootstrap particle filter ($M = 40,000$) and dotted lines correspond to conditionally-optimal particle filter ($M = 400$). Results are based on $N_{\text{run}} = 100$ runs of the filters.
Notes: Solid lines represent results from Kalman filter. Dashed lines correspond to bootstrap particle filter ($M = 40,000$) and dotted lines correspond to conditionally-optimal particle filter ($M = 400$). Results are based on $N_{run} = 100$ runs of the filters.
SW Model: Distr. of Log-Likelihood Approximation Errors

Notes: Density estimates of $\hat{\Delta}_1 = \ln \hat{p}(Y|\theta) - \ln p(Y|\theta)$ based on $N_{run} = 100$. Solid densities summarize results for the bootstrap (BS) particle filter; dashed densities summarize results for the conditionally-optimal (CO) particle filter.
Notes: Density estimates of $\hat{\Delta}_1 = \ln \hat{p}(Y|\theta) - \ln p(Y|\theta)$ based on $N_{run} = 100$. Solid densities summarize results for the bootstrap (BS) particle filter; dashed densities summarize results for the conditionally-optimal (CO) particle filter.
### SW Model: Summary Statistics for Particle Filters

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Particles $M$</td>
<td>40,000</td>
<td>400,000</td>
</tr>
<tr>
<td>Number of Repetitions</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>High Posterior Density: $\theta = \theta^m$</td>
<td></td>
</tr>
<tr>
<td>Bias $\hat{\Delta}_1$</td>
<td>-238.49</td>
<td>-118.20</td>
</tr>
<tr>
<td>StdD $\hat{\Delta}_1$</td>
<td>68.28</td>
<td>35.69</td>
</tr>
<tr>
<td>Bias $\hat{\Delta}_2$</td>
<td>-1.00</td>
<td>-1.00</td>
</tr>
<tr>
<td></td>
<td>Low Posterior Density: $\theta = \theta^l$</td>
<td></td>
</tr>
<tr>
<td>Bias $\hat{\Delta}_1$</td>
<td>-253.89</td>
<td>-128.13</td>
</tr>
<tr>
<td>StdD $\hat{\Delta}_1$</td>
<td>65.57</td>
<td>41.25</td>
</tr>
<tr>
<td>Bias $\hat{\Delta}_2$</td>
<td>-1.00</td>
<td>-1.00</td>
</tr>
</tbody>
</table>

**Notes:** Results are based on $N_{run} = 100$. 

Frank Schorfheide  
Particle Filtering
• Likelihood functions for nonlinear DSGE models can be approximated by the PF.

• We will now embed the likelihood approximation into a posterior sampler:
  • PFMH Algorithm (a special case of PMCMC)
  • $SMC^2$
• Distinguish between:
  • \{p(Y|\theta), p(\theta|Y), p(Y)\}, which are related according to:
    \[ p(\theta|Y) = \frac{p(Y|\theta)p(\theta)}{p(Y)} , \quad p(Y) = \int p(Y|\theta)p(\theta)d\theta \]
  • \{\hat{p}(Y|\theta), \hat{p}(\theta|Y), \hat{p}(Y)\}, which are related according to:
    \[ \hat{p}(\theta|Y) = \frac{\hat{p}(Y|\theta)p(\theta)}{\hat{p}(Y)} , \quad \hat{p}(Y) = \int \hat{p}(Y|\theta)p(\theta)d\theta . \]

• Surprising result (Andrieu, Docet, and Holenstein, 2010): under certain conditions we can replace \( p(Y|\theta) \) by \( \hat{p}(Y|\theta) \) and still obtain draws from \( p(\theta|Y) \).
PFMH Algorithm

For $i = 1$ to $N$:

1. Draw $\vartheta$ from a density $q(\vartheta|\theta^{i-1})$.
2. Set $\theta^{i} = \vartheta$ with probability

$$\alpha(\vartheta|\theta^{i-1}) = \min \left\{ 1, \frac{\hat{p}(Y|\vartheta)p(\vartheta)/q(\vartheta|\theta^{i-1})}{\hat{p}(Y|\theta^{i-1})p(\theta^{i-1})/q(\theta^{i-1}|\vartheta)} \right\}$$

and $\theta^{i} = \theta^{i-1}$ otherwise. The likelihood approximation $\hat{p}(Y|\vartheta)$ is computed using a particle filter.
Why Does the PFMH Work?

• At each iteration the filter generates draws \( \tilde{s}_t^j \) from the proposal distribution \( g_t(\cdot | s_{t-1}^j) \).

• Let \( \tilde{S}_t = (\tilde{s}_t^1, \ldots, \tilde{s}_t^M)' \) and denote the entire sequence of draws by \( \tilde{S}_{1:T}^1 \).

• Selection step: define a random variable \( A_t^j \) that contains this ancestry information. For instance, suppose that during the resampling particle \( j = 1 \) was assigned the value \( \tilde{s}_t^{10} \) then \( A_t^1 = 10 \). Let \( A_t = (A_t^1, \ldots, A_t^N) \) and use \( A_{1:T} \) to denote the sequence of \( A_t \)'s.

• PFMH operates on an enlarged probability space: \( \theta, \tilde{S}_{1:T} \) and \( A_{1:T} \).
Why Does the PFMH Work?

- Use $U_{1:T}$ to denote random vectors for $\tilde{S}_{1:T}$ and $A_{1:T}$. $U_{1:T}$ is an array of iid uniform random numbers.

- The transformation of $U_{1:T}$ into $(\tilde{S}_{1:T}, A_{1:T})$ typically depends on $\theta$ and $Y_{1:T}$, because the proposal distribution $g_t(\tilde{s}_t|s_{t-1}^j)$ depends both on the current observation $y_t$ as well as the parameter vector $\theta$.

- E.g., implementation of conditionally-optimal PF requires sampling from a $\mathcal{N}(\tilde{s}_{t|t}, P_{t|t})$ distribution for each particle $j$. Can be done using a prob integral transform of uniform random variables.

- We can express the particle filter approximation of the likelihood function as

$$\hat{p}(Y_{1:T}|\theta) = g(Y_{1:T}|\theta, U_{1:T}).$$

where

$$U_{1:T} \sim p(U_{1:T}) = \prod_{t=1}^{T} p(U_t).$$
Why Does the PFMH Work?

- Define the joint distribution
  \[ p_g(Y_{1:T}, \theta, U_{1:T}) = g(Y_{1:T}|\theta, U_{1:T})p(U_{1:T})p(\theta). \]
- The PFMH algorithm samples from the joint posterior
  \[ p_g(\theta, U_{1:T}|Y_{1:T}) \propto g(Y|\theta, U_{1:T})p(U_{1:T})p(\theta) \]
  and discards the draws of \((U_{1:T})\).
- For this procedure to be valid, it needs to be the case that PF approximation is unbiased:
  \[ \mathbb{E}[\hat{\theta}(Y_{1:T}|\theta)] = \int g(Y_{1:T}|\theta, U_{1:T})p(U_{1:T})d\theta = p(Y_{1:T}|\theta). \]
Why Does the PFMH Work?

- We can express acceptance probability directly in terms of $\hat{\rho}(Y_{1:T}|\theta)$.

- Need to generate a proposed draw for both $\theta$ and $U_{1:T}$: $\vartheta$ and $U^*_{1:T}$.

- The proposal distribution for $(\vartheta, U^*_{1:T})$ in the MH algorithm is given by $q(\vartheta|\theta^{(i-1)})p(U^*_{1:T})$.

- No need to keep track of the draws $(U^*_{1:T})$.

- MH acceptance probability:

$$\alpha(\vartheta|\theta^{(i-1)}) = \min \left\{ 1, \frac{g(Y|\vartheta, U^*)p(U^*)p(\vartheta)}{q(\vartheta|\theta^{(i-1)})p(U^{(i-1)})} \right\}$$

$$= \min \left\{ 1, \frac{\hat{\rho}(Y|\vartheta)p(\vartheta)/q(\vartheta|\theta^{(i-1)})}{\hat{\rho}(Y|\theta^{(i-1)})p(\theta^{(i-1)})/q(\theta^{(i-1)}|\vartheta)} \right\}.$$
• Results are based on $N_{\text{run}} = 20$ runs of the PF-RWMH-V algorithm.

• Each run of the algorithm generates $N = 100,000$ draws and the first $N_0 = 50,000$ are discarded.

• The likelihood function is computed with the Kalman filter (KF), bootstrap particle filter (BS-PF, $M = 40,000$) or conditionally-optimal particle filter (CO-PF, $M = 400$).

• “Pooled” means that we are pooling the draws from the $N_{\text{run}} = 20$ runs to compute posterior statistics.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Posterior Mean (Pooled)</th>
<th>Inefficiency Factors</th>
<th>Std Dev of Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KF</td>
<td>CO-PF</td>
<td>BS-PF</td>
</tr>
<tr>
<td>$\tau$</td>
<td>2.63</td>
<td>2.62</td>
<td>2.64</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>0.82</td>
<td>0.81</td>
<td>0.82</td>
</tr>
<tr>
<td>$\psi_1$</td>
<td>1.88</td>
<td>1.88</td>
<td>1.87</td>
</tr>
<tr>
<td>$\psi_2$</td>
<td>0.64</td>
<td>0.64</td>
<td>0.63</td>
</tr>
<tr>
<td>$\rho_r$</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>$\rho_g$</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>$\rho_z$</td>
<td>0.88</td>
<td>0.88</td>
<td>0.88</td>
</tr>
<tr>
<td>$r(A)$</td>
<td>0.44</td>
<td>0.44</td>
<td>0.44</td>
</tr>
<tr>
<td>$\pi(A)$</td>
<td>3.32</td>
<td>3.33</td>
<td>3.32</td>
</tr>
<tr>
<td>$\gamma(Q)$</td>
<td>0.59</td>
<td>0.59</td>
<td>0.59</td>
</tr>
<tr>
<td>$\sigma_r$</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
</tr>
<tr>
<td>$\sigma_g$</td>
<td>0.68</td>
<td>0.68</td>
<td>0.67</td>
</tr>
<tr>
<td>$\sigma_z$</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
</tr>
<tr>
<td>$\ln \hat{p}(Y)$</td>
<td>-357.14</td>
<td>-357.17</td>
<td>-358.32</td>
</tr>
</tbody>
</table>
Notes: The figure depicts autocorrelation functions computed from the output of the 1 Block RWMH-V algorithm based on the Kalman filter (solid), the conditionally-optimal particle filter (dashed) and the bootstrap particle filter (solid with dots).
Results are based on $N_{\text{run}} = 20$ runs of the PF-RWMH-V algorithm.

Each run of the algorithm generates $N = 10,000$ draws.

The likelihood function is computed with the Kalman filter (KF) or conditionally-optimal particle filter (CO-PF).

“Pooled” means that we are pooling the draws from the $N_{\text{run}} = 20$ runs to compute posterior statistics. The CO-PF uses $M = 40,000$ particles to compute the likelihood.
## SW Model: Accuracy of MH Approximations

<table>
<thead>
<tr>
<th></th>
<th>Post. Mean (Pooled)</th>
<th>Ineff. Factors</th>
<th>Std Dev of Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KF</td>
<td>CO-PF</td>
<td>KF</td>
</tr>
<tr>
<td>$(100\beta^{-1} - 1)$</td>
<td>0.14</td>
<td>0.14</td>
<td>172.58</td>
</tr>
<tr>
<td>$\bar{\pi}$</td>
<td>0.73</td>
<td>0.74</td>
<td>185.99</td>
</tr>
<tr>
<td>$\bar{l}$</td>
<td>0.51</td>
<td>0.37</td>
<td>174.39</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.19</td>
<td>0.20</td>
<td>149.77</td>
</tr>
<tr>
<td>$\sigma_c$</td>
<td>1.49</td>
<td>1.45</td>
<td>86.27</td>
</tr>
<tr>
<td>$\Phi$</td>
<td>1.47</td>
<td>1.45</td>
<td>134.34</td>
</tr>
<tr>
<td>$\varphi$</td>
<td>5.34</td>
<td>5.35</td>
<td>138.54</td>
</tr>
<tr>
<td>$h$</td>
<td>0.70</td>
<td>0.72</td>
<td>277.64</td>
</tr>
<tr>
<td>$\xi_w$</td>
<td>0.75</td>
<td>0.75</td>
<td>343.89</td>
</tr>
<tr>
<td>$\sigma_l$</td>
<td>2.28</td>
<td>2.31</td>
<td>162.09</td>
</tr>
<tr>
<td>$\xi_p$</td>
<td>0.72</td>
<td>0.72</td>
<td>182.47</td>
</tr>
<tr>
<td>$\iota_w$</td>
<td>0.54</td>
<td>0.53</td>
<td>241.80</td>
</tr>
<tr>
<td>$\iota_p$</td>
<td>0.48</td>
<td>0.50</td>
<td>205.27</td>
</tr>
<tr>
<td>$\psi$</td>
<td>0.45</td>
<td>0.44</td>
<td>248.15</td>
</tr>
<tr>
<td>$r_{\pi}$</td>
<td>2.09</td>
<td>2.09</td>
<td>98.32</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.80</td>
<td>0.80</td>
<td>241.63</td>
</tr>
<tr>
<td>$r_y$</td>
<td>0.13</td>
<td>0.13</td>
<td>243.85</td>
</tr>
<tr>
<td>$r_{\Delta y}$</td>
<td>0.21</td>
<td>0.21</td>
<td>101.94</td>
</tr>
<tr>
<td>Parameter</td>
<td>Post. Mean (Pooled)</td>
<td>Ineff. Factors</td>
<td>Std Dev of Means</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------</td>
<td>----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td></td>
<td>KF</td>
<td>CO-PF</td>
<td>KF</td>
</tr>
<tr>
<td>( \rho_a )</td>
<td>0.96</td>
<td>0.96</td>
<td>153.46</td>
</tr>
<tr>
<td>( \rho_b )</td>
<td>0.22</td>
<td>0.21</td>
<td>325.98</td>
</tr>
<tr>
<td>( \rho_g )</td>
<td>0.97</td>
<td>0.97</td>
<td>57.08</td>
</tr>
<tr>
<td>( \rho_i )</td>
<td>0.71</td>
<td>0.70</td>
<td>219.11</td>
</tr>
<tr>
<td>( \rho_r )</td>
<td>0.54</td>
<td>0.54</td>
<td>194.73</td>
</tr>
<tr>
<td>( \rho_p )</td>
<td>0.80</td>
<td>0.81</td>
<td>338.69</td>
</tr>
<tr>
<td>( \rho_w )</td>
<td>0.94</td>
<td>0.94</td>
<td>135.83</td>
</tr>
<tr>
<td>( \rho_{ga} )</td>
<td>0.41</td>
<td>0.37</td>
<td>196.38</td>
</tr>
<tr>
<td>( \mu_p )</td>
<td>0.66</td>
<td>0.66</td>
<td>300.29</td>
</tr>
<tr>
<td>( \mu_w )</td>
<td>0.82</td>
<td>0.81</td>
<td>218.43</td>
</tr>
<tr>
<td>( \sigma_a )</td>
<td>0.34</td>
<td>0.34</td>
<td>128.00</td>
</tr>
<tr>
<td>( \sigma_b )</td>
<td>0.24</td>
<td>0.24</td>
<td>186.13</td>
</tr>
<tr>
<td>( \sigma_g )</td>
<td>0.51</td>
<td>0.49</td>
<td>208.14</td>
</tr>
<tr>
<td>( \sigma_i )</td>
<td>0.43</td>
<td>0.44</td>
<td>115.42</td>
</tr>
<tr>
<td>( \sigma_r )</td>
<td>0.14</td>
<td>0.14</td>
<td>193.37</td>
</tr>
<tr>
<td>( \sigma_p )</td>
<td>0.13</td>
<td>0.13</td>
<td>194.22</td>
</tr>
<tr>
<td>( \sigma_w )</td>
<td>0.22</td>
<td>0.22</td>
<td>211.80</td>
</tr>
<tr>
<td>( \ln \hat{p}(Y) )</td>
<td>-964.44</td>
<td>-1017.94</td>
<td></td>
</tr>
</tbody>
</table>
We implement the PFMH algorithm on a single machine, utilizing up to twelve cores.

For the small-scale DSGE model it takes 30:20:33 [hh:mm:ss] hours to generate 100,000 parameter draws using the bootstrap PF with 40,000 particles. Under the conditionally-optimal filter we only use 400 particles, which reduces the run time to 00:39:20 minutes.

For the SW model it took 05:14:20:00 [dd:hh:mm:ss] days to generate 10,000 draws using the conditionally-optimal PF with 40,000 particles.
• Start from SMC algorithm...

• Data tempering instead of likelihood tempering: \( \pi_n^D(\theta) = p(\theta|Y_{1:t_n}) \).

• Particle filter can deliver an unbiased estimate of the incremental weight \( p(Y_{t_n-1+1:t_n} | \theta) \).

• Evaluate PF approximation of likelihood instead of true likelihood in the correction and mutation steps of SMC algorithm.

• Write:

\[
\hat{p}(y_{t_n-1+1:t_n} | Y_{1:t_n-1}, \theta) = g(y_{t_n-1+1:t_n} | Y_{1:t_n-1}, \theta, U_{1:t_n}) \\
\hat{p}(Y_{1:t_n} | \theta_n) = g(Y_{1:t_n} | \theta_n, U_{1:t_n}).
\]

• \( U_{1:t_n} \) is an array of \( iid \) uniform random variables generated by the particle filter with density \( p(U_{1:t_n}) \). Likelihood increments depend on entire \( U_{1:t_n} \). Factorization:

\[
p(U_{1:t_n}) = p(U_{1:t_1})p(U_{t_1+1:t_2}) \cdots p(U_{t_n-1+1:t_n}).
\]
Particle System for $SMC^2$ Sampler After Stage $n$

<table>
<thead>
<tr>
<th>Parameter</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\theta_1^n, W_1^n)$</td>
<td>$(s_{t_n}^{1,1}, W_{t_n}^{1,1})$</td>
</tr>
<tr>
<td>$(\theta_2^n, W_2^n)$</td>
<td>$(s_{t_n}^{2,1}, W_{t_n}^{2,1})$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$(\theta_N^n, W_N^n)$</td>
<td>$(s_{t_n}^{N,1}, W_{t_n}^{N,1})$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\theta_1^n, W_1^n)$</td>
<td>$(s_{t_n}^{1,2}, W_{t_n}^{1,2})$</td>
</tr>
<tr>
<td>$(\theta_2^n, W_2^n)$</td>
<td>$(s_{t_n}^{2,2}, W_{t_n}^{2,2})$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$(\theta_N^n, W_N^n)$</td>
<td>$(s_{t_n}^{N,2}, W_{t_n}^{N,2})$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\theta_1^n, W_1^n)$</td>
<td>$(s_{t_n}^{1,M}, W_{t_n}^{1,M})$</td>
</tr>
<tr>
<td>$(\theta_2^n, W_2^n)$</td>
<td>$(s_{t_n}^{2,M}, W_{t_n}^{2,M})$</td>
</tr>
<tr>
<td>$\vdots$</td>
<td>$\vdots$</td>
</tr>
<tr>
<td>$(\theta_N^n, W_N^n)$</td>
<td>$(s_{t_n}^{N,M}, W_{t_n}^{N,M})$</td>
</tr>
</tbody>
</table>
Initialization. Draw the initial particles from the prior: \( \theta^i_0 \overset{\text{iid}}{\sim} p(\theta) \) and \( W^0_i = 1 \), \( i = 1, \ldots, N \).

Recursion. For \( t = 1, \ldots, T \),

Correction. Reweight the particles from stage \( t - 1 \) by defining the incremental weights

\[
\tilde{w}^i_t = \tilde{p}(y_t | Y_{1:t-1}, \theta^i_{t-1}) = g(y_t | Y_{1:t-1}, \theta^i_{t-1}, U_{1:t})
\]  

and the normalized weights

\[
\tilde{W}^i_t = \frac{\tilde{w}^i_t W^i_{t-1}}{\frac{1}{N} \sum_{i=1}^{N} \tilde{w}^i_t W^i_{t-1}}, \quad i = 1, \ldots, N.
\]

An approximation of \( \mathbb{E}_{\pi_t}[h(\theta)] \) is given by

\[
\tilde{h}^{}_{t,N} = \frac{1}{N} \sum_{i=1}^{N} \tilde{W}^i_t h(\theta^i_{t-1}).
\]
1 Initialization.

2 Recursion. For $t = 1, \ldots, T$,

1 Correction.

2 Selection. Resample the particles via multinomial resampling. Let $\{\hat{\theta}_t^i\}_{i=1}^M$ denote $M$ iid draws from a multinomial distribution characterized by support points and weights $\{\theta_{t-1}^i, \tilde{W}_t^i\}_{j=1}^M$ and set $W_t^i = 1$. Define the vector of ancestors $A_t$ with elements $A_t^i$ by setting $A_t^i = k$ if the ancestor of resampled particle $i$ is particle $k$, that is, $\hat{\theta}_t^i = \theta_{t-1}^k$. An approximation of $\mathbb{E}_{\pi_t}[h(\theta)]$ is given by

$$\hat{h}_{t,N} = \frac{1}{N} \sum_{j=1}^N W_t^i h(\hat{\theta}_t^i).$$

(32)
1 Initialization.

2 Recursion. For $t = 1, \ldots, T$,
   1 Correction.
   2 Selection.

3 Mutation. Propagate the particles $\{\hat{\theta}_t^i, W_t^i\}$ via 1 step of an MH algorithm. The proposal distribution is given by

$$q(\varphi_t^i | \hat{\theta}_t^i)p(U_{1:t}^i)$$

(33)

and the acceptance ratio can be expressed as

$$\alpha(\varphi_t^i | \hat{\theta}_t^i) = \min \left\{ 1, \frac{\hat{p}(Y_{1:t}^i | \varphi_t^i)p(\varphi_t^i)}{p(\varphi_t^i)p(Y_{1:t}^i | \hat{\theta}_t^i)} \right\}.$$  (34)

An approximation of $\mathbb{E}_{\pi_t}[h(\theta)]$ is given by

$$\bar{h}_{t,N} = \frac{1}{N} \sum_{i=1}^{N} h(\theta_t^i) W_t^i.$$  (35)

3 Approximation of $\mathbb{E}_{\pi}[h(\theta)]$ is given by $\bar{h}_{T,N} = \sum_{i=1}^{N} h(\theta_T^i) W_T^i$. 

Frank Schorfheide  Particle Filtering
Why Does $SMC^2$ Work?

- At the end of iteration $t - 1$:
  - Particles $\{\theta_{t-1}^i, W_{t-1}^i\}_{i=1}^N$.
  - For each parameter value $\theta_{t-1}^i$ there is PF approx of the likelihood: $\hat{p}(Y_{1:t-1}|\theta_{t-1}^i)$.
  - Swarm of particles $\{s_{t-1}^{i,j}, W_{t-1}^{i,j}\}_{j=1}^M$ that represents the distribution $p(s_{t-1}|Y_{1:t-1}, \theta_{t-1}^i)$.
  - Sequence of random vectors $U_{1:t-1}^i$ that underlies the simulation approximation of the particle filter.

- Focus on the triplets $\{\theta_{t-1}^i, U_{1:t-1}^i, W_{t-1}^i\}_{i=1}^N$:
  \[
  \int \int h(\theta, U_{1:t-1}) p(U_{1:t-1}) p(\theta|Y_{1:t-1}) dU_{1:t-1} d\theta \\
  \approx \frac{1}{N} \sum_{i=1}^N h(\theta_{t-1}^i, U_{1:t-1}^i) W_{t-1}^i.
  \]
The particle filter approximation of the likelihood increment can be written as
\[
\hat{p}(y_t| Y_{1:t-1}, \theta_{t-1}^i) = g(y_t| Y_{1:t-1}, U_{1:t}^i, \theta_{t-1}^i).
\]

The value of the likelihood function for \( Y_{1:t} \) can be tracked recursively as follows:
\[
\hat{p}(Y_{1:t}| \theta_{t-1}^i) = \hat{p}(y_t| Y_{1:t-1}, \theta_{t-1}^i)\hat{p}(Y_{1:t-1}| \theta_{t-1}^i)
\]
\[
= g(y_t| Y_{1:t}, U_{1:t}^i, \theta_{t-1}^i)g(Y_{1:t-1}| U_{1:t-1}, \theta_{t-1}^i)
\]
\[
= g(Y_{1:t}| U_{1:t}^i, \theta_{t-1}^i).
\]

The last equality follows because conditioning \( g(Y_{1:t-1}| U_{1:t-1}, \theta_{t-1}^i) \) also on \( U_t \) does not change the particle filter approximation of the likelihood function for \( Y_{1:t-1} \).
• By induction, we can deduce that \( \frac{1}{N} \sum_{i=1}^{N} h(\theta_{t-1}^i) \tilde{w}_t^i W_{t-1}^i \) approximates the following integral

\[
\int \int h(\theta) g(y_t|Y_{1:t-1}, U_{1:t}, \theta)p(U_{1:t})p(\theta|Y_{1:t-1})dU_{1:t}d\theta = \int h(\theta) \left( \int g(y_t|Y_{1:t-1}, U_{1:t}, \theta)p(U_{1:t})dU_{1:t} \right) p(\theta|Y_{1:t-1})d\theta.
\]

• Provided that the particle filter approximation of the likelihood increment is unbiased, that is,

\[
\int g(y_t|Y_{1:t-1}, U_{1:t}, \theta)p(U_{1:t})dU_{1:t} = p(y_t|Y_{1:t-1}, \theta)
\]

for each \( \theta \), we deduce that \( \tilde{h}_{t,N} \) is a consistent estimator of \( E_{\pi_t}[h(\theta)] \).
• Similar to regular SMC.

• We resample in every period for expositional purposes.

• We are keeping track of the ancestry information in the vector $A_t$. This is important, because for each resampled particle $i$ we not only need to know its value $\hat{\theta}_t^i$ but we also want to track the corresponding value of the likelihood function $\hat{\rho}(Y_{1:t}|\hat{\theta}_t^i)$ as well as the particle approximation of the state, given by $\{s_t^{i,j}, W_t^{i,j}\}$, and the set of random numbers $U_{1:t}^i$.

• In the implementation, the likelihood values are needed for the mutation step and the state particles are useful for a quick evaluation of the incremental likelihood in the subsequent correction step.

• The $U_{1:t}^i$'s are not required for the actual implementation of the algorithm but are useful to provide a heuristic explanation for the validity of the algorithm.
Mutation Step

- Essentially one iteration of PFMH algorithm.

- For each particle $i$:
  
  - a proposed value $\vartheta^i_t$,
  
  - an associated particle filter approximation $\hat{p}(Y_{1:t}|\vartheta^i_t)$ of the likelihood,
  
  - and a sequence of random vectors $U^*_1:t$ drawn from the distribution $p(U_{1:t})$.

- The densities $p(U^i_{1:t})$ and $p(U^*_1:t)$ cancel from the formula for the acceptance probability $\alpha(\vartheta^i_t|\hat{\vartheta}^i_t)$. 

• Results are based on $N_{run} = 20$ runs of the $SMC^2$ algorithm with $N = 4,000$ particles.

• D is data tempering and L is likelihood tempering.

• KF is Kalman filter, CO-PF is conditionally-optimal PF with $M = 400$, BS-PF is bootstrap PF with $M = 40,000$. CO-PF and BS-PF use data tempering.
## Accuracy of $\text{SMC}^2$ Approximations

<table>
<thead>
<tr>
<th></th>
<th>Posterior Mean (Pooled)</th>
<th>Inefficiency Factors</th>
<th>Std Dev of Means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KF(L)</td>
<td>KF(D)</td>
<td>CO-PF</td>
</tr>
<tr>
<td>$\tau$</td>
<td>2.65</td>
<td>2.67</td>
<td>2.68</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>0.81</td>
<td>0.81</td>
<td>0.81</td>
</tr>
<tr>
<td>$\psi_1$</td>
<td>1.87</td>
<td>1.88</td>
<td>1.87</td>
</tr>
<tr>
<td>$\psi_2$</td>
<td>0.66</td>
<td>0.66</td>
<td>0.67</td>
</tr>
<tr>
<td>$\rho_r$</td>
<td>0.75</td>
<td>0.75</td>
<td>0.75</td>
</tr>
<tr>
<td>$\rho_g$</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>$\rho_z$</td>
<td>0.88</td>
<td>0.88</td>
<td>0.88</td>
</tr>
<tr>
<td>$\pi(A)$</td>
<td>3.32</td>
<td>3.31</td>
<td>3.31</td>
</tr>
<tr>
<td>$\gamma(Q)$</td>
<td>0.59</td>
<td>0.59</td>
<td>0.59</td>
</tr>
<tr>
<td>$\sigma_r$</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
</tr>
<tr>
<td>$\sigma_g$</td>
<td>0.68</td>
<td>0.68</td>
<td>0.68</td>
</tr>
<tr>
<td>$\sigma_z$</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
</tr>
<tr>
<td>$\ln p(Y)$</td>
<td>-358.75</td>
<td>-357.34</td>
<td>-356.33</td>
</tr>
</tbody>
</table>
The \(SMC^2\) results are obtained by utilizing 40 processors.

We parallelized the likelihood evaluations \(\hat{p}(Y_{1:t} | \theta^i_t)\) for the \(\theta^i_t\) particles rather than the particle filter computations for the swarms \(\{s^i_t, W^j_t\}_{j=1}^M\).

The run time for the \(SMC^2\) with conditionally-optimal PF \((N = 4,000, M = 400)\) is 23:24 [mm:ss] minutes, whereas the algorithm with bootstrap PF \((N = 4,000\) and \(M = 40,000)\) runs for 08:05:35 [hh:mm:ss] hours.

Due to memory constraints we re-computed the entire likelihood for \(Y_{1:t}\) in each iteration.

Our sequential (data-tempering) implementation of the \(SMC^2\) algorithm suffers from particle degeneracy in the initial stages, i.e., for small sample sizes.