Robust inference on parameters via particle filters and sandwich covariance matrices

Arnaud Doucet and Neil Shephard

Economics & Statistics Depts
University of Oxford
& Oxford-Man Institute

June 2012
Desire: likelihood inference on parameters of non-linear and non-Gaussian state space models.

The validity of the particle filter inside an MCMC algorithm is significant breakthrough in time series analysis. Completes the analogy of the Kalman filter to non-linear and non-Gaussian state space models. See

- Andrieu, Doucet, Holenstein (2010)

As models in economics and finance are always misspecified so likelihood inference could be entirely misleading.

Maybe better to make inference using a “sandwich matrix” adjustment.

Focus on sandwich estimation.

This needs estimates of the score for each observation. Tricky. Focus today.
Observations $y_t$ which conditioned on some states $\alpha_t$ are independent. The states are Markovian:

$$f(y_t | \alpha_t; \theta), \quad f(\alpha_t | \alpha_{t-1}; \theta),$$

where $\theta$ is a $k \times 1$ dimensional vector of parameters.

Throughout we will assume both are twice continuously differentiable with respect to $\theta$.

Reviews of the literature on these models include Harvey (1989), West and Harrison (1989), Durbin and Koopman (2001) and Cappe, Moulines and Ryden (2009).
Three forms of state estimation are very common:

- forecasting
  \[ f(\alpha_t | \mathcal{F}_{t-1}; \theta), \]
- filtering
  \[ f(\alpha_t | \mathcal{F}_t; \theta), \]
- and smoothing
  \[ f(\alpha_t | \mathcal{F}_n; \theta), \]

where \( t \leq n \).

Here \( \mathcal{F}_t \) is the information available at and including time \( t \). A side product of the filter is the prediction decomposition

\[ f(y_1, \ldots, y_n | \mathcal{F}_0; \theta) = \prod_{t=1}^{n} f(y_t | \mathcal{F}_{t-1}; \theta). \]
Except in special cases (e.g. the linear Gaussian model and the case where $\alpha_t$ only has a small number of atoms of support) these four densities have to be estimated using simulation. The leading way of carrying this out is the particle filter or sequential Monte Carlo.

Reviews of this include Cappe, Godsill and Moulines (2007), Doucet and Johansen (2009) and Creal (2011).
Particle filter: the core

- Here we recall a basic particle filter criteria:
- We start with a weighted sample of size $M$ from the $\alpha_{t-1}|\mathcal{F}_{t-1}$ which we write as $\left\{ W_{t-1}^{(i)}, \alpha_{t-1}^{(i)} \right\}$.
- We then move to a weighted sample of size $M$ from the $\alpha_{t}|\mathcal{F}_{t}$ which we write as $\left\{ W_{t}^{(i)}, \alpha_{t}^{(i)} \right\}$.
- How we do that is fascinating, but irrelevant today.
- Consider the time series

$$y_{1:n} = y_1, \ldots, y_n,$$

then particle filter delivers on the way

$$\hat{L}_\theta = \prod_{t=1}^{n} \hat{p}(y_t|\mathcal{F}_{t-1}; \theta).$$
Example

Suppose $y_t | \alpha_t \sim N(0, \sigma^2)$, $\alpha_t | \alpha_{t-1} \sim N(0, 0.1)$, $n = 100$ and $\theta = \log \sigma^2$. We draw a single path $y_{1:n}$. True value of $\theta = 0$ and plot $\log L_\theta$ and $\log \hat{L}_\theta$ as a function of $\theta$. Stratified resampling is used.
The “sample score”, if it exists, equals
\[ S_{n,\theta} = \frac{\partial \log L_\theta}{\partial \theta} = \sum_{t=1}^{n} s_{t,\theta}, \quad s_{t,\theta} = \frac{\partial l_{t,\theta}}{\partial \theta}, \quad l_{t,\theta} = \log f(y_t | \mathcal{F}_{t-1}; \theta). \]

Then a standard input into robust inference is to estimate
\[ \mathcal{I}_{n,\theta} = \text{Var} \left( n^{-1/2} \sum_{t=1}^{n} s_{t,\theta} | \mathcal{F}_0 \right) \]

Dont worry the other piece, which we also show how to estimate.
\[ \mathcal{J}_{n,\theta} = -\mathbb{E} \left( \frac{1}{n} \frac{\partial S_{n,\theta}}{\partial \theta'} | \mathcal{F}_0 \right). \]
A typical HAC estimator takes on the form

\[
\hat{\Sigma}_{n,\theta}(s) = \gamma_{n,\theta}(s; 0) + \sum_{j=1}^{P} w(j/P) \left\{ \gamma_{n,\theta}(s; j) + \gamma_{n,\theta}(s; j)' \right\},
\]

where

\[
\gamma_{n,\theta}(s; j) = \frac{1}{n} \sum_{t=j+1}^{n} (s_{t,\theta} - \bar{s}_\theta) (s_{t-j,\theta} - \bar{s}_\theta)', \quad \bar{s}_\theta = \frac{1}{n} \sum_{t=1}^{n} s_{t,\theta}.
\]

Here \(w\) is a weight function.

- Then under very weak conditions, which involve \(P\) increasing very slowly with \(n\) discussed extensively in the above literature, 
  \(\hat{\Sigma}_{n,\theta}(s) - \mathcal{I}_{n,\theta} \xrightarrow{p} 0\) as \(n \to \infty\).
Estimating the individual scores

Note that

\[ S_n = \sum_{t=1}^{n} E_{\alpha_t|F_n} \left\{ \frac{\partial \log f(y_t|\alpha_t; \theta)}{\partial \theta} \right\} + \sum_{t=2}^{n} E_{\alpha_t,\alpha_{t-1}|F_n} \left\{ \frac{\partial \log f(\alpha_t|\alpha_{t-1}; \theta)}{\partial \theta} \right\} \]

\[ + E_{\alpha_1|F_n} \left\{ \frac{\partial \log f(\alpha_1|F_0; \theta)}{\partial \theta} \right\} . \]

e.g. Koopman & Shephard (1992) and Shephard (1993).

But it does not give us the time series of individual scores for

\[ s_t = \frac{\partial \log f(y_t|F_{t-1}; \theta)}{\partial \theta} = S_t - S_{t-1} \]

\[ \neq E_{\alpha_t|F_n} \left\{ \frac{\partial \log f(y_t|\alpha_t; \theta)}{\partial \theta} \right\} + E_{\alpha_t|F_n} \left\{ \frac{\partial \log f(\alpha_t|\alpha_{t-1}; \theta)}{\partial \theta} \right\} . \]

Harvey (1989): recursion for individual scores for linear models.

Needed for general state spaces in order to compute our HAC.
Del Moral, Doucet and Singh (2011) have a sequential estimator of $S_{t,\theta}$. Our target is somewhat different, $s_{t,\theta}$ and the corresponding sandwich estimator, but we will piggyback on their work. So let us recall their approach.
Let us ignore the initial condition, then construct

\[ u_t(\alpha_t, \alpha_{t-1}) = \frac{\partial \log f(y_t|\alpha_t; \theta)}{\partial \theta} + \frac{\partial \log f(\alpha_t|\alpha_{t-1}; \theta)}{\partial \theta}, \]

\[ U_T(\alpha_1:T) = \sum_{t=1}^{n} u_t(\alpha_t, \alpha_{t-1}), \]

\[ S_t(\alpha_t) = \int U_t(\alpha_{1:t})dF(\alpha_{1:t-1}|\mathcal{F}_{t-1}, \alpha_t). \]

Now the sample score is

\[ S_t = E_{\alpha_t|\mathcal{F}_t}\{S_t(\alpha_t)\}. \]
Del Moral, Doucet and Singh (2011) see that

\[ S_t(\alpha_t) = \mathbb{E}_{\alpha_{t-1}|\mathcal{F}_{t-1},\alpha_t} \{ S_{t-1}(\alpha_{t-1}) + u_t(\alpha_t, \alpha_{t-1}) \} . \] (1)

They sequentially computing \( S_t(\alpha_t) \) and then off those \( S_t \).

Now define the functional

\[
    s_t(\alpha_t) = S_t(\alpha_t) - S_{t-1} = \mathbb{E}_{\alpha_{t-1}|\alpha_t,\mathcal{F}_{t-1}} [ u_t(\alpha_t, \alpha_{t-1}) ] \\
    + \mathbb{E}_{\alpha_{t-1}|\alpha_t,\mathcal{F}_{t-1}} \{ S_{t-1}(\alpha_{t-1}) - S_{t-2} \} + (S_{t-2} - S_{t-1}) \\
    = \mathbb{E}_{\alpha_{t-1}|\alpha_t,\mathcal{F}_{t-1}} [ u_t(\alpha_t, \alpha_{t-1}) + s_{t-1}(\alpha_{t-1}) - s_{t-1} ] .
\]

Then

\[ s_t = \mathbb{E}_{\alpha_t|\mathcal{F}_t} \{ s_t(\alpha_t) \} . \]

Task: estimate via simulation the functionals \( s_{t-1}(\alpha_{t-1}) \).
Particle implementation

Functional recursion

\[ S_t(\alpha_t) = E_{\alpha_{t-1}|F_{t-1},\alpha_t} \{ S_{t-1}(\alpha_{t-1}) + u_t(\alpha_t, \alpha_{t-1}) \} \]

\[ = \frac{\int f(\alpha_t|\alpha_{t-1}) \{ S_{t-1}(\alpha_{t-1}) + u_t(\alpha_t, \alpha_{t-1}) \} \, dF(\alpha_{t-1}|F_{t-1})}{\int f(\alpha_t|\alpha_{t-1}) \, dF(\alpha_{t-1}|F_{t-1})}. \]

Del Moral, Doucet and Singh (2011) suggested estimating this by

\[ \hat{S}_t(\alpha_t) = \sum_{i=1}^{M} W_{t-1}^{(i)} f(\alpha_t|\alpha_{t-1}^{i}) \left\{ S_{t-1}(\alpha_{t-1}^{i}) + u_t(\alpha_t, \alpha_{t-1}^{i}) \right\} \]

\[ \left( \sum_{i=1}^{M} W_{t-1}^{(i)} f(\alpha_t|\alpha_{t-1}^{i}) \right), \]

where \( \{ W_{t-1}^{(i)}, \alpha_{t-1}^{i} \} \) are from \( dF(\alpha_{t-1}|F_{t-1}) \). Forward only. Then

\[ \hat{S}_t = \sum_{j=1}^{M} W_t^{(j)} \hat{S}_t(\alpha_{t|t-1}^{j}). \] (2)

But \( O(M^2) \). Resample before scoring, remove weights reduce \( M \).
Example

Plot $S_{n, \theta}$ and $\hat{S}_{n, \theta}$ as a function of $\theta = \log \sigma^2$, again for a single draw of $y_{1:n}$. We vary $M = 100, 250, 1,000$ and $2,500$. Each particle filter estimator of the score is stochastically independent for each value of $\theta$. 
An individual scores version

The same approach can be applied to the time series of individual scores

\[
    s_t(\alpha_t) = \frac{\int f(\alpha_t|\alpha_{t-1}) \{s_{t-1}(\alpha_{t-1}) + u_t(\alpha_t, \alpha_{t-1})\} \, dF(\alpha_{t-1}|\mathcal{F}_{t-1})}{\int f(\alpha_t|\alpha_{t-1}) \, dF(\alpha_{t-1}|\mathcal{F}_{t-1})} - s_{t-1}.
\]

This would deliver

\[
    \hat{s}_t(\alpha_t) = -s_{t-1} + \left( \sum_{i=1}^{M} W_t^{(i)} f(\alpha_t|\alpha_{t-1}^i) \right)^{-1} \times \sum_{i=1}^{M} W_t^{(i)} f(\alpha_t|\alpha_{t-1}^i) \{s_{t-1}(\alpha_{t-1}^i) + u_t(\alpha_t, \alpha_{t-1}^i)\}.
\]

Then

\[
    \hat{s}_t = \sum_{j=1}^{M} W_t^{(j)} \hat{s}_t(\alpha_{t|t-1}^j).
\]
Example

Plot $s_{t,\theta}$ against $\hat{s}_{t,\theta}$ as a function of $t = 1, 2, \ldots, n$ taking $\theta = 0$, the true value and $n = 100$. We vary $M = 100, 250, 1,000$ and $2,500$. The results show a tightening of the estimator as $M$ increases.
Extension of earlier work on the sample score by Del Moral, Doucet and Singh (2011).

Theorem: Assume Assumption A, given in the Appendix. For any $r > 1$, there exists a constant $c_r < \infty$ such that for any $\theta \in \Theta$, $y = \{y_t\}_{t \geq 0}$, $t \geq 0$, that as $M \to \infty$

$$\sqrt{M} \left\{ E_{t, \theta}^M |\hat{s}_t - s_t|^r \right\}^{1/r} < c_r.$$ 

Here the expectations are over the particles of size $M$ using the parameter $\theta$, conditioning on $y$.

- Note it is possible to derive a Gaussian central limit theory for $\sqrt{M} (\hat{s}_t - s_t)$, as $M \to \infty$, with uniformly bounded asy var. However, the resulting asy var is of not a great deal of practical importance beyond it being uniformly bounded.
Example

The simulation based estimation of $s_{t, \theta}$. Top we show the 0.1 and 0.9 quantiles of the estimation errors of $s_{t, \theta} - \hat{s}_{t, \theta}$. 

![Graphs showing the simulation based estimation of $s_{t, \theta}$ for different $M$ values.](chart)

- (a) $M = 100$
- (b) $M = 250$
- (c) $M = 1000$
- (d) $M = 2500$
Theorem

Assume Assumption A, given in the Appendix. Then for all $\theta \in \Theta$, $y = \{y_t\}_{t \geq 0}$, as $M \to \infty$ so

$$HAC(\hat{s}) - HAC(s) \overset{u.p.}{\to} 0$$

so long as $P / M \to 0$. 
Overcoming computational burden

For the first time the particle approach needs an $O(M^2)$ step. Computing

$$
\left( \sum_{i=1}^{M} f(\alpha_t | \alpha_{t-1}^i) \right)^{-1} \times \sum_{i=1}^{M} f(\alpha_t | \alpha_{t-1}^i) \left\{ s_{t-1}(\alpha_{t-1}^i) + u_t(\alpha_t, \alpha_{t-1}^i) \right\},
$$

for every value of $\alpha_t^i$. This is boring in practice and has impact. Hence desire for parallelisation.

- Arnaud has argued to me that simple Matlab parallelisation is sufficient for this to be doable in practice without too much pain.
- It has been a significant pain without parallelisation.
Jurgen Doornik has allowed me to use the new version of Ox which has an additional command to supplement

- for \((i=0; i<N; i++)\)

Now also has

- parallel for \((i=0; i<N; i++)\)

This splits up the work onto the multiple cores of the processors. Only change in the code. Of course MPI type things can also be used on many systems. Jurgen is currently researching on accessing graphics code within Ox using openCL.

Parallel for can only be used for “safe calculations” (non-iterative), easy to do silly things and get the wrong answer. But rules are easy too.
My desktop has two Intel(r) Xeon(r) X5660 (Six Core, 2.80 GHz, 12MB Cache, 6.40 GT/s Intel(r) QPI).


<table>
<thead>
<tr>
<th>M</th>
<th>100</th>
<th>250</th>
<th>1,000</th>
<th>2,000</th>
<th>5,000</th>
<th>10,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>time/M (secs)</td>
<td>0.0018</td>
<td>0.0013</td>
<td>0.0026</td>
<td>0.0054</td>
<td>0.018</td>
<td>0.026</td>
</tr>
</tbody>
</table>

**Table:** Parallel for case: Times per particle divided by M

<table>
<thead>
<tr>
<th>M</th>
<th>100</th>
<th>250</th>
<th>1,000</th>
<th>2,000</th>
<th>5,000</th>
<th>10,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>time/M (secs)</td>
<td>0.0045</td>
<td>0.0086</td>
<td>0.035</td>
<td>0.063</td>
<td>0.14</td>
<td>0.29</td>
</tr>
</tbody>
</table>

**Table:** Serial for case: Times per particle divided by M

Roughly linearly up to $M = 1,000$ and is roughly quadratic.

Gain over existing Oxmetrics: factor of 10 on the same machine.
Estimating the Hessian via particles

To estimate robust standard errors we have to estimate

\[ \mathcal{J}_{n,\theta} = -E \left( n^{-1} \sum_{t=1}^{n} \frac{\partial^2 l_{t,\theta}}{\partial \theta \partial \theta'} | \mathcal{F}_0 \right). \]

We do this by replacing the expectation with an average. The immediate task is thus to compute the Hessian

\[ \Delta^2 l_{1:n} = \frac{\partial^2 \log p_{\theta}(y_{1:n})}{\partial \theta \partial \theta'}. \]

The approach we follow is numerically equivalent to Poyiadjis, Doucet and Singh (2011) but removes some unnecessary computations and has a simpler derivation.
First we set

\[ U_t = \frac{\partial \log p_\theta(y_{1:t}, \alpha_{1:t})}{\partial \theta} = U_{t-1} + u_t(\alpha_t, \alpha_{t-1}), \]

\[ V_t = \frac{\partial^2 \log p_\theta(y_{1:t}, \alpha_{1:t})}{\partial \theta \partial \theta'} = V_{t-1} + v_t(\alpha_t, \alpha_{t-1}), \quad (6) \]

\[ v_t(\alpha_t, \alpha_{t-1}) = \frac{\partial^2 \{ I(y_t|\alpha_t) + I(\alpha_t|\alpha_{t-1}) \}}{\partial \theta \partial \theta'}. \quad (7) \]

Now note that

\[ \Delta^2 l_{1:t} = \frac{\partial^2 \log p_\theta(y_{1:t})}{\partial \theta \partial \theta'} = \frac{\partial}{\partial \theta'} \left( \frac{1}{p_\theta(y_{1:t})} \frac{\partial p_\theta(y_{1:t})}{\partial \theta} \right) = B_t - S_t S_t', \]

where

\[ B_t = \frac{1}{p_\theta(y_{1:t})} \frac{\partial^2 p_\theta(y_{1:t})}{\partial \theta \partial \theta'}, \quad S_t = \frac{\partial \log p_\theta(y_{1:t})}{\partial \theta}, \]

and using the Louis (1982) formula,

\[ B_t = H_t + K_t, \quad (8) \]

\[ H_t = \int \{ U_t(\alpha_{1:t}) U(\alpha_{1:t})' \} \, dF(\alpha_{1:t} | \mathcal{F}_t), \quad (9) \]

\[ K_t = \int V(\alpha_{1:t}) \, dF(\alpha_{1:t} | \mathcal{F}_t), \quad (10) \]
So define the functionals, suppressing dependence on $\alpha_{1:t}$,

\[ H_t(\alpha_t) = \int (U_t U_t') \, dF(\alpha_{1:t-1}|\mathcal{F}_{t-1}, \alpha_t), \]
\[ S_t(\alpha_t) = \int U_t \, dF(\alpha_{1:t-1}|\mathcal{F}_{t-1}, \alpha_t), \]
\[ K_t(\alpha_t) = \int V_t \, dF(\alpha_{1:t-1}|\mathcal{F}_{t-1}, \alpha_t), \]

delivers the desired

\[ H_t = E_{\alpha_t|\mathcal{F}_t} \{ H_t(\alpha_t) \}, \quad S_t = E_{\alpha_t|\mathcal{F}_t} \{ S_t(\alpha_t) \}, \quad K_t = E_{\alpha_t|\mathcal{F}_t} \{ K_t(\alpha_t) \}. \]
Obviously $U_t U'_t = (U_{t-1} + u_t) (U_{t-1} + u_t)'$ and
$E_{\alpha_{1:t-1}|\mathcal{F}_{t-1},\alpha_t} = E_{\alpha_{t-1}|\mathcal{F}_{t-1},\alpha_t} E_{\alpha_{1:t-2}|\mathcal{F}_{t-2},\alpha_{t-1}}$, so we run in parallel

\[
H_t(\alpha_t) = E_{\alpha_{t-1}|\mathcal{F}_{t-1},\alpha_t} \left\{ H_{t-1}(\alpha_{t-1}) + u_t u'_t + S_{t-1}(\alpha_{t-1}) u'_t \\ + u_t S_{t-1}(\alpha_{t-1})' \right\},
\]

\[
S_t(\alpha_t) = E_{\alpha_{t-1}|\mathcal{F}_{t-1},\alpha_t} \left\{ S_{t-1}(\alpha_{t-1}) + u_t \right\},
\]

\[
K_t(\alpha_t) = E_{\alpha_{t-1}|\mathcal{F}_{t-1},\alpha_t} \left\{ K_{t-1}(\alpha_{t-1}) + v_t \right\}.
\]
Particle implementation

We will implement this using the existing weighted particles \( \{ W_{t-1}^{(j)}, \alpha_{t-1}^{(j)} \} \) (as before, in practice, it computationally makes sense to resample to make the weights equal before carrying this out, but we ignore that here). Write

\[
H_{j,t-1} = H_{t-1}(\alpha_{t-1}^{(j)}), \quad S_{j,t-1} = S_{t-1}(\alpha_{t-1}^{(j)}), \quad K_{j,t-1} = K_{t-1}(\alpha_{t-1}^{(j)}),
\]

\[
u_{j,t}(\alpha_t) = u_t(\alpha_t, \alpha_{t-1}^{(j)}), \quad v_{j,t}(\alpha_t) = v_t(\alpha_t, \alpha_{t-1}^{(j)}), \quad f_{j,t}(\alpha_t) = f(\alpha_t | \alpha_{t-1}^{(j)})
\]

and, again, generically

\[
\hat{E}_{j|\alpha_t} (H_{j,t-1}) = \frac{\sum_{j=1}^{M} W_{t-1}^{(j)} f(\alpha_t | \alpha_{t-1}^{(j)}) H_{j,t-1}}{\sum_{j=1}^{M} W_{t-1}^{(j)} f(\alpha_t | \alpha_{t-1}^{(j)})}.
\]
Then

\[
\hat{H}(\alpha_t) &= \hat{E}_{j|\alpha_t} \left\{ H_{j,t-1} + u_{j,t}(\alpha_t)u'_{j,t}(\alpha_t) + S_{j,t-1}u'_{j,t}(\alpha_t) + u_{j,t}(\alpha_t)S'_{j,t-1} \right\}, \\
\hat{S}(\alpha_t) &= \hat{E}_{j|\alpha_t} \left\{ S_{j,t-1} + u_{j,t}(\alpha_t) \right\}, \\
\hat{K}(\alpha_t) &= \hat{E}_{j|\alpha_t} \left\{ K_{j,t-1} + v_{j,t}(\alpha_t) \right\}.
\]

This drives

\[
\hat{H}_t = \sum_{j=1}^M W_t^{(j)} \hat{H}_t(\alpha_t^{(j)}), \quad \hat{S}_t = \sum_{j=1}^M W_t^{(j)} \hat{S}_t(\alpha_t^{(j)}), \quad \hat{K}_t = \sum_{j=1}^M W_t^{(j)} \hat{K}_t(\alpha_t^{(j)}).
\]

The resulting estimator is thus

\[
\Delta^2 \hat{l}_1:t = \hat{H}_t - \hat{S}_t\hat{S}'_t + \hat{K}_t.
\]
Monte Carlo assessment of the estimator of robust standard errors

We now move onto assessing the performance of this approach in cases where the model is incorrect.

Example

Suppose our model is still $y_t | \alpha_t; \theta \sim N(\alpha_t, \sigma^2)$ and $\alpha_t | \alpha_{t-1} \sim N(\alpha_{t-1}, 1)$ and $\theta = \log \sigma^2$, but now the data generating process is taken to be

$$y_t = \alpha_t + \varepsilon_t,$$

$$\varepsilon_t \sim \frac{\chi^2_p - p}{\sqrt{2p}}, \quad p > 0,$$

where $\varepsilon_t$ is i.i.d.. Measurement error is highly skewed for small $p$. As we are fitting a linear model the quasi-likelihood with deliver a consistent estimator of the pseudo-true value of $\theta$, which is $\theta^* = \log \text{Var}(\varepsilon_t) = 0$. The values of $p$ vary through $\{1, 2, 3, 4, 5, 10, 25\}$, where $p = 25$ delivers a law for $\varepsilon_t$ which is close to being Gaussian. We taken $M$ through $\{100, 250, 1000, 2500\}$ and $n = 100$ and $n = 250$. 
<table>
<thead>
<tr>
<th>quantile</th>
<th>0.5</th>
<th>0.5</th>
<th>0.99</th>
<th>0.99</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>100</td>
<td>250</td>
<td>100</td>
<td>250</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$M$</th>
<th>$p = 1$</th>
<th>$p = 2$</th>
<th>$p = 3$</th>
<th>$p = 4$</th>
<th>$p = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>6.2</td>
<td>5.4</td>
<td>5.6</td>
<td>5.2</td>
<td>4.5</td>
</tr>
<tr>
<td>250</td>
<td>4.6</td>
<td>3.1</td>
<td>3.0</td>
<td>2.8</td>
<td>3.1</td>
</tr>
<tr>
<td>1,000</td>
<td>2.2</td>
<td>1.7</td>
<td>1.7</td>
<td>1.6</td>
<td>1.6</td>
</tr>
<tr>
<td>2,500</td>
<td>1.3</td>
<td>1.1</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
</tr>
</tbody>
</table>
Multicores allows us to do different calculation, e.g. $O(M^2)$ ones.
The individual scores can be estimated for general state space models.
The HAC estimator based upon estimated scores is consistent.
The Hessian can be approximated by particle methods.
Allows robust standard errors for state space models.