An introduction to Sequential Monte Carlo Techniques

CAEPR Joint Macro-Econometrics Minicourse

Gianni Amisano (FRB, UTS)

CAEPR Joint Macro-Econometrics Minicourse, IU

2015/09/23

2015/09/23

1Views expressed here are not those of the Federal Reserve System
Nonlinear state space model

Dynamic system

(measurement equation) \( y_t^O = G(x_t, v_t, \theta) \) \hspace{1cm} (1)

(state equation) \( x_t = H(x_{t-1}, w_t, \theta) \) \hspace{1cm} (2)
Filtering problem

- projection

\[ p(x_{t+1}|y^o_t, \theta) = \int p(x_{t+1}|x_t, \theta)p(x_t|y^o_t, \theta) dx_t \]  

- update

\[ p(x_{t+1}|y^o_{t+1}, \theta) = \frac{p(x_{t+1}|y^o_t, \theta)p(y^o_{t+1}|x_{t+1}, \theta)}{p(y^o_{t+1}|y^o_t, \theta)} \]  

\[ p(y^o_{t+1}|y^o_t, \theta) = \int p(x_{t+1}|y^o_t, \theta)p(y^o_{t+1}|x_{t+1}, \theta) dx_{t+1} \]  

Integration steps easy only under very special circumstances (KF, Hamilton filter are examples)
Sequential Monte Carlo (SMC) methods

- Arulampalam *et al.* (2002), IEEE
- Doucet *et al.* (2001)
- Fernandez-Villaverde and Rubio-Ramirez (2007)
- An and Schorfheide (2007)
- Durbin and Koopman (2010)
- Durham and Geweke (2014)

⇒ Filtering by simulation.
Simplest way: Particle Filter (PF)
Intuition of the PF

compute the likelihood \( p(y_{t+1}^o | y_t^o, \theta) \) by:

1. drawing large number of realisations from distribution of \( x_{t+1} \) conditioned on \( y_t^o \)
2. assigning them weight determined by their "distance" from (compatibility with) \( y_{t+1}^o \).

For Bayesians:
\[
p(x_{t+1} | y_t^o, \theta) = \text{prior distribution (prior to observing } y_{t+1}^o)\]
\[
p(y_{t+1}^o | x_{t+1}, \theta) = \text{"likelihood"},
\]
\( \Rightarrow \) doing posterior simulation drawing from prior and using likelihood as weights.
Suppose we have $N$ draws to approximate $p(x_t|y^o_t, \theta)$ (swarm of particles):

$$
\left( x_t^{(i)}, w_t^{(i)} \right), \ i = 1, 2, \ldots, N
$$

(6)

Weight $w_t^{(i)}$ in case $x_t^{(i)}$ drawn from $q(x_t)$ (Importance sampling):

$$
w_t^{(i)} = \frac{p(x_t^{(i)}|y^o_t, \theta)}{q(x_t^{(i)})}
$$

(7)
Two ways to compute expected value of any function $f$ of $x_t$:

- **direct Importance Sampling (IS):**

  $$E \left[ f(x_t^{(i)}) \mid y_{t}, \theta \right] \approx \frac{\sum_{i=1}^{N} w_t^{(i)} f(x_t^{(i)})}{\sum_{i=1}^{N} w_t^{(i)}} \quad (8)$$

- **Resample** $x_t^{(i)}$ drawing $N$ times from empirical distribution of the $x_t^{(i)}$, with probabilities $w_t^{(i)}$ ⇒

  $$\left( x_t^{(j)}, 1 \right), j = 1, 2, \ldots, N \quad (9)$$

  $$E \left[ f(x_t^{(i)}) \mid y_{t}, \theta \right] \approx \frac{\sum_{j=1}^{N} f(x_t^{(j)})}{N} \quad (10)$$
Filtering, I

Assuming to have a swarm of particles with perfectly even weights ($w_t^{(j)} = 1, j = 1, 2, ..., N$):

- projection

\[
p(x_{t+1}|y_{t+1}^o, \theta) \approx \frac{1}{N} \sum_{j=1}^{N} p(x_{t+1}|x_t^{(j)})
\]

(11)

Empirically performed by drawing $x_{t+1}^{(j)}$ from

\[
p(x_{t+1}|x_t^{(j)}, \theta)
\]

(12)

(i.e. simulate state equation) \( \left( x_{t+1}^{(j)}, 1 \right), j = 1, 2, ..., N \)
Filtering, II

Update: drawn from $p(x_{t+1} | y_{t}^{o}, \theta)$ but wanted to draw from $p(x_{t+1} | y_{t+1}^{o}, \theta) \Rightarrow$ assign weights proportional to $p(y_{t+1}^{o} | x_{t+1}^{(j)}, \theta)$.

Updated distribution $p(x_{t+1} | y_{t+1}^{o}, \theta)$ is approximated by the sample

$$\left( x_{t+1}^{(j)}, w_{t+1}^{(j)} \right), j = 1, 2, ..., N,$$

$$w_{t+1}^{(j)} = p(y_{t+1}^{o} | x_{t+1}^{(j)}, \theta)$$ (14)

This sample can be resampled using the weights $w_{t+1}^{(j)}$ as probabilities.
Resample or not resample? I

If not resampling at each step, then weights will accumulate

$$
\prod_{i=0}^{t-1} w_{t-i}^{(j)} = \prod_{i=0}^{t-1} p(y_{t-i}^o | x_{t-i}^{(j)})
$$

After a while (at some $t$) weight assigned to the particle even marginally most compatible with the observable data will be 1 and all the others will be zero $\Rightarrow$ numerical accuracy of the filter quickly deteriorates.
Resample or not resample? II

In any case, better to watch your weights!

NEFF, (i.e. numerical efficiency index):

\[ NEFF_t = \sum_{i=1}^{N} \left( w_t^{(i)} \right)^2 \]  \hspace{0.5cm} (16)

\( \approx \) Herfindhal-Hirschmann index. Has to be safely far from 1 and as close as possible to \( 1/N \)
Computation of the likelihood

Sample mean of unnormalised weights (14) is $t^{th}$ observation likelihood conditional on past values of observables:

$$\frac{1}{N} \sum_{j=1}^{N} p(y_{t+1}^{o}|x_{t+1}^{(j)}, \theta)$$

$$\approx \int \int p(y_{t+1}^{o}|x_{t+1}, \theta) p(x_{t+1}|x_{t}, \theta) p(x_{t}|y_{t}^{o}, \theta) dx_{t+1} dx_{t} =$$

$$= p(y_{t+1}^{o}|y_{t}^{o}, \theta)$$

(17)

$\Rightarrow$ Likelihood based inference (Bayesian or not)

Inference on unobservables (smoothed or filtered)
Importance function $q$ should be more spread out than target distribution $p$: bounded weights $w_t^{(i)} = \frac{p_i}{q_i}$.
But if IS distribution too spread out, large number of draws given negligible weights $\Rightarrow$ poor numerical accuracy properties.
$\Rightarrow$ PF is based on a blind proposal
Figure 1: first particle will be killed either by reweighting or by resampling.
Importance sampling (IS) interpretation, II

Figure 1
Sensitivity to outliers

see Figure 2: particle 3 will get a unit weight
No or low measurement error

\[ p(y_{t+1}^o|x_{t+1}^{(j)}, \theta) \] to compute weights, but if no measurement error this becomes degenerate.
Simplest way: use SMC to produce likelihood

- to be maximised or
- to be combined with prior to obtain posterior, via MCMC (Metropolis-Hastings)
MH algorithm

random walk Metropolis Hastings algorithm (see Chib, 2001) which works by sequentially repeating the following steps:

- draw $\theta^{(i)}$ from a candidate distribution $q_V(\theta^{(i-1)})$;
- compute the solution of the DSGE model and the implied state space form;
- carry out the simulation filter which will produce also the likelihood of the model

$$p(y^o_T | \theta^{(i)}) = \prod_{t=1}^{T-1} p(y^o_{t+1} | y^o_t, \theta^{(i)});$$

- accept $\theta^{(i)}$ with probability

$$\frac{p(\theta^{(i)}) p(y^o_T | \theta^{(i)})}{p(\theta^{(i-1)}) p(y^o_T | \theta^{(i-1)})}$$ (18)

if the draw is not accepted the MH simulator sets $\theta^{(i)} = \theta^{(i-1)}$. 

Papers to read

- Durham and Geweke (2014)
- Herbsts and Schorfheide (2012), (2016 forthcoming)

(So far used with models where likelihood is available analytically)
A simple example: dynamic pooling of models

- Del Negro, Hasegawa, Schorfheide (2015)
- Amisano and Geweke (2013, 2015)
A simple example of SMC at work

Forecasting combination

- 3 models (DFM, DSGE and VAR), producing predictive densities
  \[ p(y_{t+1}|y_{1:t}, M_i), \ i = 1, 2, 3 \]

- Model combination
  \[ p(y_{t+1}|y_{1:t}) = \sum_{i=1}^{3} p(y_{t+1}|y_{1:t}, M_i) w_t \]
Density forecasting combination

- Focus on density forecasts
- Produce density forecast based on model combination, formula
- A possibility is to use Bayesian Model Averaging, with weights proportional to the posterior
- Problem: this approach assumes that one of the combined model is true
- Asymptotically, least wrong model is selected with weight equal to one
- This asymptotic tends to kick in quite fast
An alternative: Optimal pooling (I)

- Use optimal weights
- Use weights to maximise log predictive density of linear combination of model predictive densities

\[
wt_{t-1} = \arg \max \sum_{s=1}^{t-1} \ln \left[ \sum_{i=1}^{n} w_i p(y_s | y_{1:s-1}, M_i) \right]
\]

- Originally proposed by Hall and Mitchell (2007), as way to find minimum KL distance to unknown DGP.
- Way to address misspecification
An alternative: optimal pooling (II)

- Sensible way of combining models when none is true
- Model value: drop model j from the pool and compute difference in log scores of the combination
- This can be decomposed period by period (each month) or by subperiods (recessions/expansions)
- measure also contribution during different recessions: not all recessions are driven by financial factors
Problems with optimal pooling

• Optimal weights are imprecisely estimated

• We might want to have a method capable of dynamically giving more weights to models that might be particularly suited for certain circumstances (eg. crisis and its immediate aftermath)
  
  • Analogy with stockpicking techniques

• Very difficult to beat equal weights (1/N), if N number of models
A simple example of SMC at work

The method in the DHS paper (I)

- 3 models producing predictive densities

\[ p(y_t | y_{1:t-1}, M_i), i = 1, 3 \]

- Combine them with weight \( w_t \) (here scalar, just two models) evolving in time as

\[
\zeta_t = \rho \zeta_{t-1} + \sqrt{(1 - \rho^2)} \times e_t, \quad e_t \sim NID(0, I_3), \tag{19}
\]

\[
\omega_{it} = \frac{\exp(\zeta_{it})}{\sum_{j=1}^m \exp(\zeta_{jt})}, \quad i = 1, 2, 3. \tag{20}
\]

- Neat:
  - when \( \rho = 1 \), we are back to static weights
  - prior for the weights is uniform on \([0,1]\)
  - Other approaches much more complicated
The method in the DHS paper (II)

- Use Sequential Monte Carlo (particle filter) to simulate the densities

\[ p(\xi_t | y_{1:t}) \] (filtered)
\[ p(\xi_t | y_{1:t-1}) \] (predictive)

- Use these to combine models
- In particular, use predictive weights for real time evaluations of pools
Out of sample approach

All RT approaches are constructed by using recursively information up to t-1 to compute performance at t. The algorithm works as follows

- For t=1,2,..., T compute
  \[ p(y_{1:t} | \rho_r) = \prod_{t=1}^{T} p(y_t | \Omega_{t-1}, \rho_r), \ r = 1, 2, .., R \]
  for a fine grid of values of \( \rho \)
- identify \( \rho_{r^*,t} \)
- compute
  \[ p(y_{1:t} | \Omega_0, \rho_{r^*,t-1}) = \prod_{t=1}^{T} p(y_t | \Omega_{t-1}, \rho_{r^*,t-1}), \ r = 1, 2, .., R \]
Results

A simple example of SMC at work

Figure: Real time optimal pool weights using dynamic pooling