ESAIM: PROCEEDINGS, September 2007, Vol.19, 12-17
Christophe Andrieu & Dan Crisan, Editors
DOI: 10.1051/proc:071903

PARTICLE FILTERING FOR CONTINUOUS-TIME HIDDEN MARKOV MODELS [*]

Nicolas Chopin[1] and Elisa Varini[2]

Abstract. We consider continuous-time models where the observed process depends on an unobserved jump Markov Process. We develop a sequential Monte Carlo algorithm which makes it possible to filter and smooth this latent process, and compute the likelihood pointwise. We develop a Rao-Blackwellisation technique which allows to significantly reduce the Monte Carlo noise of this algorithm. Possible extensions of our algorithm and further directions of research are discussed.

1. Model and notations

We consider a continuous-time model involving an unobserved pure jump Markov process (X_t) with finite state space $\mathcal{S} = \{1, \dots, S\}$, initial distribution $\phi_a(0)$ at time 0, $a = 1, ., S$, and generator matrix Q,

$$(Q)_{k,l} = \begin{cases} -q_k & \text{if } k = l, \\ q_k q_{kl} & \text{otherwise.} \end{cases}$$

Thus, the times between jumps are exponentially distributed, with rate q_k when in state k, and the jumps from k to l occur with probability q_{kl}.

The observed process is denoted (Y_t), and the observation interval is $[0, T]$; this process is characterised by its conditional likelihood $L^c(Y_{[0,T]}, X_{[0,T]})$, with respect to the appropriate dominating measure and for a given realisation $X_{[0,T]}$. Specifically, we assume a conditional independent model, that is, Y_t is independent of $X_{t'}$, $t' \neq t$, conditional on X_t, for any $t \in [0, T]$. For convenience, the notation $L^c(Y_{[0,T]}, X_{[0,T]})$ is shortened to $L^c(X_{[0,T]})$ in the rest of the paper, as $Y_{[0,T]}$ remains constant. Two simple examples are conditional point process models with intensity function $\lambda(t, X_t)$, conditional on (X_t), and switching linear diffusions models such as those discussed in [Liechty and Roberts, 2001]. In these applications, $L^c(X_{[0,T]})$ can often be computed analytically, and we assume this is the case in the following.

One wishes to filter (X_t) at times $0 < t_1 < \dots < t_N$; say $t_0 = 0$ and $t_N = T$. A chain rule decomposition of L^c yields

$$L^c(X_{[0,T]}) = \prod_{n=0}^{N-1} L^c_n(X_{(t_n,t_{n+1}]})$$

where $L^c_n(X_{(t_n,t_{n+1}]})$ stands for the likelihood of $Y_{(t_n,t_{n+1}]}$, conditional on the observed past $Y_{(t_n,t_{n+1}]}$, and on $X_{[0,t_{n+1}]}$, or equivalently $X_{[t_n,t_{n+1}]}$.

[*] *We are grateful to Tobyas Ryden for helpful comments. Part of this work was done while first author was supported by EPSRC grant EP/C015886/1.*

[1] ENSAE, France, and Bristol University, United Kingdom.
[2] CNR, Milano, Italy.

If (Y_t) is a point process with conditional intensity $\lambda(t, X_t)$, its observation can be summarised by occurrence times $\tau_1 < \ldots < \tau_N$, and

$$L^c(X_{[0,T]}) = \prod_{n=1}^{N} \lambda(\tau_n, X_{\tau_n}) \exp\left\{ -\int_0^T \lambda(s, X_s)\, ds \right\}.$$

One would typically filter at times $t_n = \tau_n$, then

$$L_n^c(X_{(t_n,t_{n+1}]}) = \lambda(t_{n+1}, X_{t_{n+1}}) \exp\left\{ -\int_{t_n}^{t_{n+1}} \lambda(s, X_s)\, ds \right\}$$

but other choices are possible.

The vector θ of unknown parameters contain the elements of Q, plus possibly a sub-vector ξ of parameters attached to $L^c(\cdot)$. For simplicity, we assume that the initial state distribution $(\phi_k(0))$ of (X_t) do not depend on θ; e.g. $\phi_k(0) = 1/S$ for $k = 1, \ldots, S$.

1.1. **Theoretical forward recursion**

For any time $t \in (t_n, t_{n+1}]$, the filtering probabilities $\phi_k(t)$ are derived in [Varini, 2005]:

$$\phi_k(t) = \varphi_k(t) / \sum_{l=1}^{S} \varphi_l(t),$$

where

$$\varphi_k(t) = E^{(n)}\left[L^c_{(t_n,t]}(X_{(t_n,t]}) I(X_t^{(n)} = k) \right], \tag{1}$$

and $E^{(n)}[\cdot]$ denotes the expectation with respect to a jump Markov process $X_t^{(n)}$ on the interval $(t_n, t_{n+1}]$, with generator Q, and initial distribution $\phi_k(t_n) = \varphi_k(t_n) / \sum_{l=1}^{S} \varphi_l(t_n)$.

The likelihood of interval $(t_n, t_{n+1}]$, conditional on X_{t_n}, is given by

$$L_{(t_n,t_{n+1}]} = \sum_{k=1}^{S} \varphi_k(t_{n+1}), \tag{2}$$

and the complete likelihood is obtained by multiplying recursively (2).

1.2. **Particle filtering**

A first possible Monte Carlo implemention of the filtering recursions above is as follows:

Repeat, for $n = 0, \ldots, N-1$,

(1) simulate $H^{(n)} = \sum_{a=1}^{S} H_a^{(n)}$ realisations $(\widetilde{X}_t^{(n,j)})$ of a jump Markov process defined on $[t_n, t_{n+1}]$, with generator Q, and starting in state 1 at time t_n for the $H_1^{(n)}$ first realisations, in state 2 for the next $H_2^{(n)}$ realisations, etc.
(2) For $j = 1, \ldots, H^{(n)}$, denote a the starting state of particle j, and compute particle weight

$$w^{(n,j)} = \frac{\widehat{\phi}_a(t_n)}{H_a^{(n)}} L^c(\widetilde{X}^{(j)}_{(t_n,t_{n+1}]}). \tag{3}$$

(3) For $k = 1, \dots, S$, compute

$$\widehat{\varphi}_k(t_{n+1}) = \sum_{j=1}^{H^{(n)}} w^{(n,j)} I(\widetilde{X}_{t_{n+1}}^{(n,j)} = k),$$

and $\widehat{\phi}_k(t_{n+1}) = \widehat{\varphi}_k(t_{n+1}) / \sum_{l=1}^{S} \widehat{\varphi}_l(t_{n+1})$, $\widehat{L}_{(t_n,t_{n+1}]} = \sum_{k=1}^{S} \widehat{\varphi}_k(t_{n+1})$.

Note that simulating a jump Markov process is straightforward, as times between jumps are exponentially distributed, with rate q_a when in state a, and state transitions are Markov, with probabilities q_{ab} for $b \neq a$.

A distinctive feature of our algorithm is the absence of a resampling step, where particles are selected randomly according to their weights, and carried forward in the next iteration. Since X_{t_n} is discrete-valued, we can decide instead that $H_a^{(n)}$ particles start in state a at time t_n, for $a = 1, \dots, S$; the relative weight of the respective subpopulation are maintained by the ratio $\widehat{\phi}_a(t_n)/H_a^{(n)}$ in (3). A reasonable choice for the $H_a^{(n)}$'s seems to be $H_a^{(n)} = \lceil H\widehat{\phi}_a(t_n)\rceil$, so that $H \leq H^{(n)} \leq H + S$, where H is chosen prior to the execution.

1.3. **Particle smoothing**

We describe here a smoothing algorithm that produces, up to some asymptotically negligible error, H' draws from the posterior distribution of $X_{[0,T]}$, conditional on $Y_{[0,T]}$ and on θ. This second algorithm takes as an input the particle values and weights generated during every iteration of the filtering algorithm above. Interestingly, the cost of our smoothing algorithm is $O(\max(H, H'))$, while Monte Carlo smoothing usually costs $O(HH')$ [Godsill et al., 2004]. The simulated paths $\overline{X}_{[0,T]}^{(j)}$ are constructed backwards, starting with $\overline{X}_{[0,T]}^{(j)}$:

(1) Perform resampling on the set of particles $\widetilde{X}_{[t_{N-1},T]}^{(N-1,j)}$, with respect to their weights $w^{(N-1,j)}$, $j = 1, \dots, H$, so as to obtain H' resampled paths, denoted $\overline{X}_{[t_{N-1},T]}^{(j)}$, $j = 1, \dots, H'$. Let $n \leftarrow N - 2$.

(2) Let $n_i = \sum_{j=1}^{H'} I[\overline{X}_{t_{n+1}}^{(j)} = i]$, for $i = 1, \dots, S$ and $k \leftarrow 1$. For $i = 1, \dots, S$, perform the following operations: resample the subset of particles $\widetilde{X}_{[t_n,t_{n+1}]}^{(n,j)}$ such that $\widetilde{X}_{t_{n+1}}^{(n,j)} = i$, with respect to their weights $w^{(n,j)}$, so as to obtain n_i resampled paths, denoted $\overline{X}_{[t_n,t_{n+1}]}^{(j)}$, for $k \leq j \leq k + n_i - 1$; then increment k by n_i.

(3) $n \leftarrow n - 1$. If $n > 0$ go to (2).

Although this is rarely done in practice, all of the resampling schemes we are aware of (e.g. multinomial resampling, [Gordon et al., 1993]; residual resampling, [Liu and Chen, 1998]; systematic resampling, [Kitagawa, 1996], [Carpenter et al., 1999]) allow for producing H' resampled particles from H weighted particles, even if $H' \neq H$. Residual and systematic resampling are to be preferred, as they lead to smaller Monte Carlo errors.

2. Rao-Blackwellisation

2.1. **Decomposition**

Our Rao-Blackwellisation scheme is based on the following decomposition of $\varphi_k(t_{n+1})$, as defined in (1). Denote N_J the number of jumps of (X_t) in $(t_n, t_{n+1}]$, $S_0 = X_{t_n}$, and S_1, S_2 the values taken by (X_t) after first and second jump, respectively, provided these jumps occur.

$$\begin{aligned}\varphi_k(t_{n+1}) &= E\left[E^{(n)}\left[L^c(X_{[t_n,t_{n+1}]})I(X_{t_{n+1}}^{(n)} = k) \mid N_J,\ S_0,\ S_1,\ S_2\right]\right] \\ &= \varphi_k^0(t_{n+1}) + \varphi_k^1(t_{n+1}) + \varphi_k^{2+}(t_{n+1}) \qquad (4)\end{aligned}$$

where

$$\varphi_k^0(t_{n+1}) = P(N_J = 0, S_0 = k) E^{(n)}\left[L^c(X_{[t_n,t_{n+1}]}) \mid N_J = 0,\ S_0 = a\right]$$

corresponds to event 'no jumps in $(t_n, t_{n+1}]$',

$$\varphi_k^1(t_{n+1}) = \sum_{a\neq k}^{S} P(N_J = 1, S_0 = a, S_1 = k) \\ E^{(n)}\left[L^c(X_{[t_n,t_{n+1}]}) \mid N_J = 1,\ S_0 = a,\ S_1 = k\right]$$

corresponds to event 'one jump exactly in $(t_n, t_{n+1}]$', and

$$\varphi_k^{2+}(t_{n+1}) = \sum_{a\neq b, b\neq c} P(N_J > 1,\ S_0 = a,\ S_1 = b,\ S_2 = c) \\ E^{(n)}\left[L^c(X_{[t_n,t_{n+1}]}) I(X_t^{(n)} = k) \mid N_J > 1,\ S_0 = a,\ S_1 = b,\ S_2 = c\right] \quad (5)$$

to 'two jumps or more'. Due to particular properties of exponential distributions, these expressions simplify as follows: (Details of calculations are omitted for sake of space.)

$$\varphi_k^0(t_{n+1}) = \phi_k(t_n) e^{-q_k(t_{n+1}-t_n)} L^c(k),$$

where $L^c(k)$ is L^c evaluated at constant function $s \to k$, for $s \in (t_n, t_{n+1}]$,

$$\varphi_k^1(t_{n+1}) = \sum_{a\neq k}^{S} \phi_a(t_n) q_{ak} e^{-q_k(t_{n+1}-t_n)} \\ \int_{t_n}^{t_{n+1}} e^{-(q_a-q_k)(t_c-t_n)} L^c(a1_{s<t_c} + k1_{s\geq t_c})\, dt_c, \quad (6)$$

and $P(N_J > 1,\ S_0 = a,\ S_1 = b, S_2 = c) = \phi_a(t_n) q_{ab} q_{bc} \varepsilon_n(a,b)$, where

$$\varepsilon_n(a,b) = 1 + \frac{1}{q_b - q_a}\left\{q_a e^{-q_b(t_{n+1}-t_n)} - -q_b e^{-q_a(t_{n+1}-t_n)}\right\}. \quad (7)$$

Depending on the complexity of the considered model, a closed-form expression for the integral in (6) may not be available; but since this integral is one-dimensional and has compact support, it can be evaluated using numerical quadrature methods. The only part of the decomposition for which it makes sense to use Monte Carlo is the expectation in the term corresponding to 'two jumps or more'; see (5).

2.2. Rao-Blackwellised filter and smoother

Our improved particle filter algorithm reads as follows:

Repeat, for $n = 0, \ldots, N-1$,

(1) Set $j = 1$, then increment j by one each time a new particle is created in the following: for $a, b, c = 1, \ldots, S$, $a \neq b$, $b \neq c$, simulate $H_{abc}^{(n)}$ realisations $(\widetilde{X}_t^{(n,j)})$ of a jump Markov process defined on $[t_n, t_{n+1}]$, with generator Q, and conditioned on starting in state a, jumping at least twice before t_{n+1}, the two first jumps being to state b, then to state c. Let $\overline{H}^{(n)} = \sum H_{abc}^{(n)}$.

(2) For $j = 1, \ldots, H^{(n)}$, let (a, b, c) be the three first states visited by particle $(\widetilde{X}_t^{(n,j)})$, and assign weight

$$w^{(n,j)} = \frac{\widehat{\phi}_a(t_n) q_{ab} q_{bc} \varepsilon_n(a,b)}{H_{abc}^{(n)}} L^c_{(t_n,t_{n+1}]}(\widetilde{X}^{(n,j)}_{(t_n,t_{n+1}]}),$$

to particle j, where $\varepsilon_n(a,b)$ is given by (7).

(3) Add the following weighted particles to the particle system, and increment the particle index j accordingly:
   (a) for $a = 1, \dots, S$, set $X_{(n,j)}$ to constant function $s \to a$, with weight

$$w^{(n,j)} = \widehat{\phi}_a(t_n) e^{-q_a(t-t_n)} L^c_{(t_n,t_{n+1}]}(a),$$

   (b) for $a, b = 1, \dots, S$, $a \neq b$, set $X_{(n,j)}$ to $(a1_{s<t_c} + b1_{s\geq t_c})$ but without specifying the value of t_c (i.e. $X_{(n,j)}$ represents a trajectory that jumps exactly once, from a to b, at an unknown time) and assign weight

$$w^{(n,j)} = \widehat{\phi}_a(t_n) q_{ab} e^{-q_b(t_{n+1}-t_n)} \int_{t_n}^{t_{n+1}} e^{-(q_a-q_b)(t_c-t_n)} L^c(a1_{s<t_c} + b1_{s\geq t_c})\, dt_c.$$

(4) Set $H^{(n)} = \overline{H}^{(n)} + S^2$; for $k = 1, \dots, S$, compute

$$\widehat{\varphi}_k(t_{n+1}) = \sum_{j=1}^{H^{(n)}} w^{(n,j)} I(\widetilde{X}^{(n,j)}_{t_{n+1}} = k),$$

$\widehat{\phi}_k(t_{n+1}) = \widehat{\varphi}_k(t_{n+1}) / \sum_{l=1}^S \widehat{\varphi}_l(t_{n+1})$, and $\widehat{L}_{(t_n,t_{n+1}]} = \sum_{k=1}^S \widehat{\varphi}_k(t_{n+1})$.

In (1), simulating a jump Markov process conditioned on jumping at least twice in $[t_n, t_{n+1}]$, from a to b, then from b to c, amounts to draw the two first times between jumps g_1, g_2 from the joint density of two exponentials, constrained to have their sum being smaller than $t_{n+1} - t_n$, then to simulate a standard jump Markov process over $[t_n + g_1 + g_2, t_{n+1}]$ with starting value c at time $t_n + g_1 + g_2$. The simulation of these two first times by a simple accept-reject algorithm, the details of which are omitted for sake of space.

Note that in (c.ii), for the particles corresponding to the event 'exactly one jump in $[t_n, t_{n+1}]$, the jump time is unknown (and is marginalised out in the corresponding weight), but this is fine since successive computations only depend on $X_{t_{n+1}}$.

If one replaces the first filtering algorithm by its rao-blackwellised version, the smoothing strategy described in §1.3 can still be applied to the output of the algorithm. The only modification required is that, when a particle is selected in the backward construction that corresponds to the event 'exactly one jump in $[t_n, t_{n+1}]$', the time t_c when this single jump occurs must be simulated from the appropriate truncated exponential distribution.

The particle sample sizes $H^{(n)}_{abc}$ may be set to: $H^{(n)}_{abc} = \lceil H q_{ab} q_{bc} \varepsilon_n(a, b) \rceil$, so that the Monte Carlo effort scales with the probability associated to the considered term of the likelihood function; the number of particles $H^{(n)}$ is then at most $H + S(S-1)^2 + S^2$.

3. Discussion and extensions

The numerical experiments we have performed so far seem to indicate that the rao-blackwellised algorithm leads to very small Monte Carlo errors; e.g. 10^-5 relative accuracy for likelihood function, for $H = 60$, whereas the naive filter needs $H = 6 \times 10^4$ particles to attain 10^-3 relative accuracy. As already said, the Monte Carlo part of our improved algorithm corresponds to an event with small probability, that is that (X_t) jumps twice or more in a small interval.

We briefly mention possible extensions. One objective is to evaluate the gradient of the likehood function, which requires extended calculations of derivatives of the various terms of decomposition 4. A second objective is to force the algorithm to produce a 'smooth' evaluation of the likelihood and its gradient for different parameter values. In this way, standard optimisation techniques, which typically do not work with noisy functions, can be used for maximum likelihood estimation. We have developed a strategy that achieves this, and which is based on some bookkeeping and recycling of the same random numbers for different evaluations. We also have derived a Monte Carlo EM algorithm, whose Monte Carlo part relies on our smoothing algorithm, and we are currently investigating possible MCMC strategies for Bayesian inference.

References

[Carpenter et al., 1999] Carpenter, J., Clifford, P., and Fearnhead, P. (1999). Improved particle filter for nonlinear problems. *IEE Proc. Radar, Sonar Navigation*, 146(1):2–7.

[Godsill et al., 2004] Godsill, S. J., Doucet, A., and West, M. (2004). Monte Carlo smoothing for nonlinear time series. *J. Am. Statist. Assoc.*, 99:156–168.

[Gordon et al., 1993] Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F, Comm., Radar, Signal Proc.*, 140(2):107–113.

[Kitagawa, 1996] Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comput. Graph. Statist.*, 5:1–25.

[Liechty and Roberts, 2001] Liechty, J. C. and Roberts, G. O. (2001). Markov chain monte carlo methods for switching diffusion models. *Biometrika*, 88:299–315.

[Liu and Chen, 1998] Liu, J. and Chen, R. (1998). Sequential Monte Carlo methods for dynamic systems. *J. Am. Statist. Assoc.*, 93:1032–1044.

[Varini, 2005] Varini, E. (2005). Sequential estimation methods in continuous-time state-space models. *PhD thesis, Institute of Quantitative Methods, Bocconi University.*