# Monte Carlo Filtering of Piecewise-Deterministic Processes

Adam M. Johansen
with Nick Whiteley and Simon J. Godsill

University of Warwick, Dept. of Statistics
a.m.johansen@warwick.ac.uk

EPSRC Workshop on MCMC and Related Methods

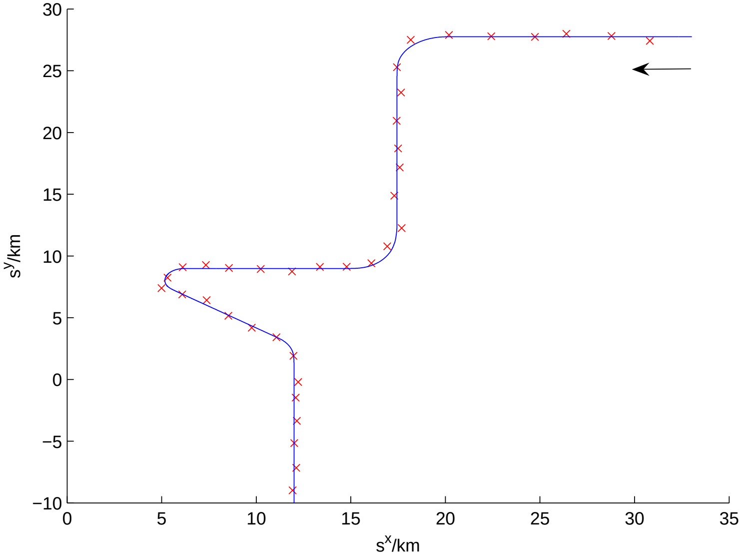
# Outline

- Background
  - Piecewise Deterministic Processes (PDPs)
  - Sequential Monte Carlo (Samplers)
- Methodology
  - "Particle Filtering" of PDPs
  - "Auxiliary Particle Filtering" of PDPs
- Examples
  - Object Tracking
  - Rate Estimation: Shot Noise Cox Processes

# Motivation: Observing a Manoeuvering Object

- For $t \in \mathbb{R}_0^+$, consider object with position $s_t$, velocity $v_t$ and acceleration $a_t$
- Summarise state by $\zeta_t = (s_t, v_t, a_t)$
- From initial condition $\zeta_0$, state evolves until random time $\tau_1$, at which acceleration jumps to a new random value, yielding $\zeta_{\tau_1}$
- From $\zeta_{\tau_1}$, evolution until $\tau_2$, state becomes $\zeta_{\tau_2}$, etc.
- Observation times, $(t_n)_{n \in \mathbb{N}}$, at each $t_n$ a noisy measurement of the object's position is made

# Example Trajectory



# An Abstract Formulation

- Pair Markov chain $(\tau_j, \theta_j)_{j \in \mathbb{N}}$, $\tau_j \in \mathbb{R}^+$, $\theta_j \in \Theta$

$$p(d(\tau_j, \theta_j)|\tau_{j-1}, \theta_{j-1}) = q(d\theta_j|\theta_{j-1}, \tau_j, \tau_{j-1})f(d\tau_j|\tau_{j-1}),$$

- Count the jumps $\nu_t := \sum_j \mathbb{I}_{[\tau_j \leq t]}$
- Deterministic evolution function $F : \mathbb{R}_0^+ \times \Theta \to \Theta$, s.t. $\forall \theta \in \Theta$,

$$F(0, \theta) = \theta$$

- Signal process $(\zeta_t)_{t \in \mathbb{R}_0^+}$,

$$\zeta_t := F(t - \tau_{\nu_t}, \theta_{\nu_t})$$

- This describes a piecewise deterministic process.
- It's partially observed via observations $(Y_n)_{n \in \mathbb{N}}$, with likelihood function $g_n(y_n|\zeta_{t_n})$

# Sequential Importance Resampling

For a sequence, $\pi_n$, of distributions over spaces $E^n$:

- Set $n = 1$
- For $i = 1 : N$
  - Sample $X_1^i \sim q_1$.
  - Calculate $W_1^i \propto \pi_1(X_1^i)/q_1(X_1^i)$.
- Iterate: $n \leftarrow n + 1$
  - Resample $(W_{n-1}^i, X_{n-1}^i)$ to obtain $(1/N, X_{n,1:n-1}^i)$.
  - For $i = 1 : N$
    - Sample $X_{n,n}^i \sim q_n(\cdot|X_{n,1:n-1}^i)$.
    - Calculate

$$W_n^i \propto \frac{\pi_n(X_{n,1:n}^i)}{\pi_{n-1}^i(X_{n,1:n-1}^i) q_n(X_{n,n}^i|X_{n,1:n-1}^i)}$$

In a filtering context, can use $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$.

# The Variable Rate Particle Filter

The VRPF iteration:

- Resample $(W_{n-1}^i, (k_{n-1}^i, \tau_{1:k_{n-1}^i}^i, \theta_{1:k_{n-1}^i}^i))$.
- For $i = 1 : N$
  - While $\tau_{n,k_n}^i < t_n$
    - Sample $\tau_{n,k_n+1}^i \sim f(\cdot|\tau_{n,k_n}^i)$.
    - $k_n^i \leftarrow k_n^i + 1$.
    - Sample $\theta_{n,k_n^i}^i \sim h(\cdot|\tau_n, \theta_{n,k_{n-1}^i}^i)$.
  - Calculate

$$W_n^i \propto \frac{g(y_n|x_n^i) f(\theta_{k_{n-1}^i+1:k_n^i}|\theta_{k_{n-1}^i})}{h(\theta_{k_{n-1}^i+1:k_n^i}^i|\tau_{n,1:k_{n-1}^i}^i, \theta_{n,k_{n-1}^i}^i)}$$

Something like the bootstrap filter.

# SMC Samplers

SMC can be used to sample from *any* sequence of distributions (Del Moral et al., 2006).

- Given *target* distributions, $\eta_n$, on $E_n \ldots$,
- construct a synthetic sequence $\widetilde{\eta}_n$ on spaces $\bigotimes_{p=1}^{n} E_p$
- by introducing Markov kernels, $L_p$ from $E_{p+1}$ to $E_p$:

$$\widetilde{\eta}_n(x_{1:n}) = \eta_n(x_n) \prod_{p=1}^{n-1} L_p\left(x_{p+1}, x_p\right),$$

- These distributions
  - have the target distributions as time marginals,
  - have the correct structure to employ SMC techniques.

# Inference and PDPs

- Consider the spaces $(E_n)_{n \in \mathbb{N}}$,

$$E_n = \biguplus_{k=0}^{\infty} \{k\} \times \mathbb{T}_{n,k} \times \Theta^{k+1}$$

$$\mathbb{T}_{n,k} = \{\tau_{1:k} : 0 < \tau_1 < \tau_2 < \ldots < \tau_k \leq t_n\}.$$

- Define $k_n := \nu_{t_n}$ and $X_n = (\zeta_0, k_n, \tau_{1:k_n}, \theta_{1:k_n}) \in E_n$
- Sequence of posterior distributions $(\eta_n)_{n \in \mathbb{N}}$

$$\eta_n(x_n) \propto q(\zeta_0) \prod_{j=1}^{k_n} f(\tau_j|\tau_{j-1}) q(\theta_j|\theta_{j-1}, \tau_j, \tau_{j-1})$$

$$\times \prod_{p=1}^{n} g_p(y_p|\zeta_{t_p}) S(\tau_{k_n}, t_n)$$

# Particle Filtering of PDPs

- Can use SMC sampler to target $\eta_n$ for $n = 1, \ldots$.
- This allows more sophisticated proposals:
  - Use of observations.
  - Explicit dimension-changing proposals.
  - No sampling into the future.
- Often interested only in $p(\zeta_{t_n} | y_{1:n})$.
- Consequently, need only keep track of $(k_n, \tau_{k_n}, \theta_{k_n})$.
- Many techniques from particle filtering can be used.

# Algorithmic Details

- Use a mixture of moves:
  - *Birth* moves.
  - *Refinement* moves.
- Augment with MCMC moves if required:
  - Can include RJMCMC moves.
- Use observations to obtain good proposals.
- Recover the VRPF under certain circumstances.

# Auxiliary Particle Filtering of PDPs

- Can construct an analogue of discrete time APF:
  - Use $n + 1^{\text{st}}$ observation at time $n$.
  - Prevents resampling eliminating promising particles.
  - Pre-weighting is then corrected for by later weights.
  - Interpretation:
    - Approximate $\hat{p}(\cdot|y_{1:n+1})$ at time $n$.
    - Correct with weights $p(\cdot|y_{1:n})/\hat{p}(\cdot|y_{1:n+1})$.

# Algorithmic Details

- An auxiliary PDP particle filter comprises:
  - SMC sampler for auxiliary sequence$(\mu_n)_{n \geq 1}$,
  - Sequence of importance weights, $(\widetilde{W}_n)$ between $(\mu_n)_{n \geq 1}$ and $(\eta_n)_{n \geq 1}$.
- We use auxiliary distributions of the following form:

$$\mu_n(x_n) \propto V_n(\tau_{n,k_n}, \theta_{n,k_n}, y_{n+1})\eta_n(x_n)$$

  with $V_n : \mathbb{R}^+ \times \Xi \to (0, \infty)$.
- $V_n$ approximates predictive likelihood.

# Back to the motivating example...

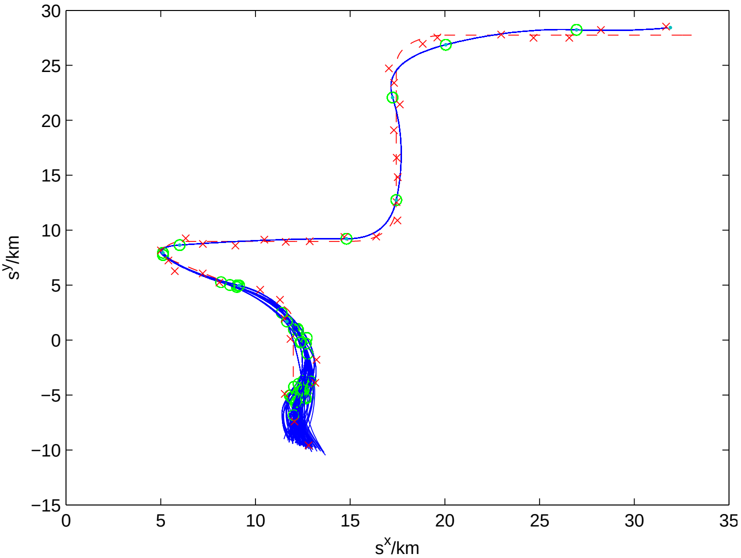
- Observation interval, $\Delta = 5$ s.
- Changepoint intervals are *a priori* Gamma$(a, b)$ with:

$$a = 10 \qquad\qquad b = 5\Delta/a$$

- Used a mixture of birth and adjustment moves (in proportion 2:1).
- Observations are noisy range/bearing pairs.
- Model linearisation used to approximate optimal kernels.
- Stratified resampling when ESS $< N/2$.

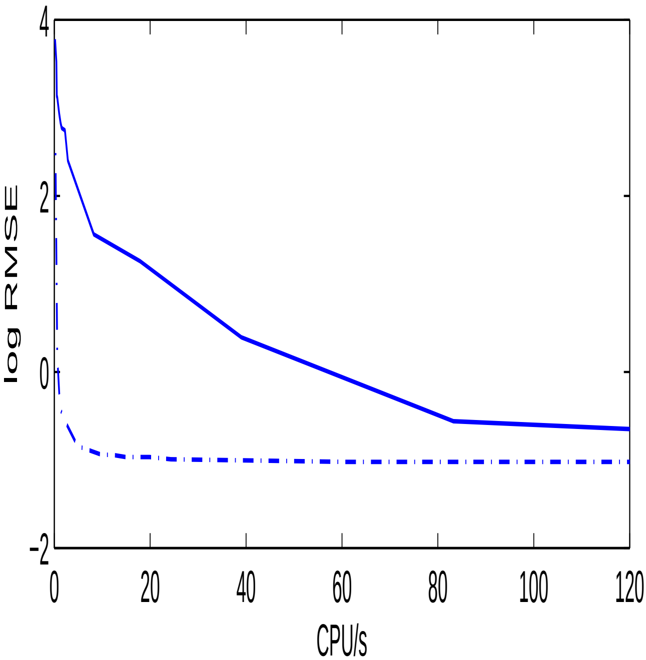
- Can also consider a Rao-Blackwellised version.

# The final sample set



| $N$ | Godsill et al. 2007 RMSE / km | Godsill et al. 2007 CPU / s | Whiteley et al. 2007 RMSE / km | Whiteley et al. 2007 CPU / s |
|---|---|---|---|---|
| 50 | 42.62 | 0.24 | 0.88 | 1.32 |
| 100 | 33.49 | 0.49 | 0.66 | 2.62 |
| 250 | 22.89 | 1.23 | 0.54 | 6.56 |
| 500 | 17.26 | 2.42 | 0.51 | 12.98 |
| 1000 | 12.68 | 5.00 | 0.50 | 26.07 |
| 2500 | 6.18 | 13.20 | 0.49 | 67.32 |
| 5000 | 3.52 | 28.79 | 0.48 | 142.84 |

Root mean square filtering error and
CPU time — over 200 runs.


4
2
0
-2
0
20
40
60
80
100
120
CPU/s
log RMSE

# Estimation of Insurance Event Rates

The model is specified by the following distributions:

$$f(\tau_j|\tau_{j-1}) = \lambda_\tau \exp(-\lambda_\tau(\tau_j - \tau_{j-1})) \times \mathbb{I}_{[\tau_{j-1},\infty)}(\tau_j),$$
$$q(\zeta_0) = \exp(-\lambda_\theta \zeta_0) \times \mathbb{I}_{[0,\infty)}(\zeta_0),$$
$$q(\theta_j|\theta_{j-1},\tau_j,\tau_{j-1}) = \lambda_\theta \exp(-\lambda_\theta(\theta_j - \zeta_{\tau_j}^-)) \times \mathbb{I}_{[\zeta_{\tau_j}^-,\infty)}(\theta_j),$$

where

$$\zeta_{\tau_j}^- = \theta_{j-1} \exp(-\kappa(\tau_j - \tau_{j-1})), \text{ and}$$
$$F(t-\tau,\theta) = \theta \exp(-\kappa(t-\tau)).$$

The likelihood function is given by:

$$g(y_n|\zeta_{(t_{n-1},t_n]}) = \exp\left(-\int_{t_{n-1}}^{t_n} \zeta_s \mathrm{d}s\right) \prod_i \zeta_{y_{n,i}}$$

where $y_{n,i}$ is the time of the $i$th event observed in $(t_{n-1}, t_n]$.

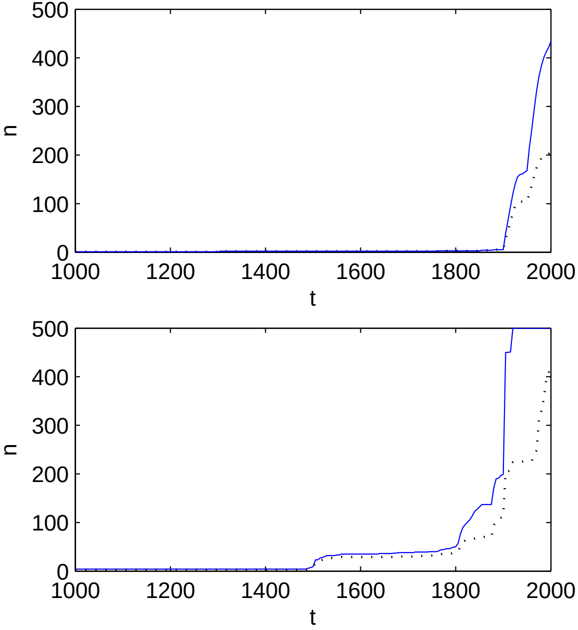
# Implementation Details

- Proposal step: simple birth proposal.

$$K_n(x_{n-1}, dx_n) = \delta_{k_{n-1}+1}(k_n)\delta_{\tau_{n-1,1:k_{n-1}}}(d\tau_{n,1:k_n-1})$$
$$\times \delta_{\theta_{n-1,1:k_{n-1}}}(d\theta_{n,1:k_n-1})h_n(d\tau_{n,k_n})$$
$$\times \pi_n(d\theta_{n,k_n}|x_n \setminus \theta_{n,k_n}),$$

- Systematic resampling applied when ESS $< 0.4N$.
- RJMCMC move applied.

# Unique Particles



# SMCTC: A C++ Template Class

- Implementing SMC algorithms in C/C++ isn't hard.

- Software for implementing general SMC algorithms.
- C++ element largely confined to the library.
- Available (under a GPL-3 license from)

  `www2.warwick.ac.uk/fac/sci/statistics/staff/academic/johansen/smctc/`

  or type "smctc" into google.
- Particle filters can also be implemented easily.

# Conclusions

- SMC sampler-type approaches allow efficient online algorithms to be developed.
- The proposed approach can dramatically outperform existing algorithms.
- This approach allows a class of piecewise deterministic models to be used when online inference is required.
- The VRPF can be interpreted within this framework, allowing existing convergence results to be applied to it.

- Possible extensions:
  - Parameter estimation via PMCMC.

# References


P. Del Moral, A. Doucet, and A. Jasra.
Sequential Monte Carlo samplers.
*Journal of the Royal Statistical Society B*, 63(3):411–436, 2006.

S. J. Godsill, J. Vermaak, K.-F. Ng, and J.-F. Li.
Models and algorithms for tracking of manoeuvring objects using variable rate particle filters.
*Proceedings of IEEE*, 95(5):925–952, 2007.

A. M. Johansen.
SMCTC: Sequential Monte Carlo in C++.
Research Report 08:16, University of Bristol, Department of Mathematics – Statistics Group, University Walk, Bristol, BS8 1TW, UK, July 2008.

N. Whiteley, A. M. Johansen, and S. Godsill.
Efficient Monte Carlo filtering for discretely observed jumping processes.
In *Proceedings of IEEE Statistical Signal Processing Workshop*, pages 89–93, Madison, WI, USA, August 26th–29th 2007. IEEE.

N. Whiteley, A. M. Johansen, and S. Godsill.
Monte Carlo filtering of piecewise-deterministic processes.
*In revision.*, 2008.