Divide-and-Conquer sequential Monte Carlo with applications to high dimensional filtering

Francesca R. Crucinio

11 July 2023

SMC Down Under

Outline

A reminder on SMC

Sequential Monte Carlo (SMC) methods approximate sequences of targets $\{\hat{\eta}_n\}_{n\geq 1}$ of the form $\hat{\eta}_n = \hat{\gamma}_n / Z_n$ with

$$\begin{aligned}\hat{\gamma}_n(x_{1:n}) &= \prod_{t=1}^{n} M_t(x_{t-1}, x_t) G_t(x_t) \\ &= \hat{\gamma}_{n-1}(x_{1:n-1}) M_n(x_{n-1}, x_n) G_n(x_n)\end{aligned}$$

where M_t are Markov kernels and G_t are non-negative weight functions.

Can be used

- as an alternative to MCMC (Del Moral et al., 2006)
- for rare event simulation (Cérou et al., 2012)
- for ABC (Sisson et al., 2007; Del Moral et al., 2012)
- for filtering

State Space Models (SSM)

$$
\begin{tikzcd}
 & Y_{t-1} & Y_t & \\
\cdots \arrow[r] & X_{t-1} \arrow[u] \arrow[r, "f_t"] & X_t \arrow[u, "g_t"'] \arrow[r] & \cdots
\end{tikzcd}
$$

A state space model $(X_t, Y_t)_{t\geq 1} \subset \mathbb{R}^{d\times p}$ with

- transition density for the *latent* process $f_t(x_{t-1}, x_t)$
- likelihood for the *observation* process $g_t(x_t, y_t)$

SSM

In this case

$$\begin{aligned}
\hat{\gamma}_n(x_{1:n}) &= p(x_{1:n}, y_{1:n}) \\
&= \prod_{t=1}^{n} f_t(x_{t-1}, x_t) g_k(x_t, y_t) \\
&= p(x_{1:n-1}, y_{1:n-1}) f_n(x_{n-1}, x_n) g_n(x_n, y_n)
\end{aligned}$$

and

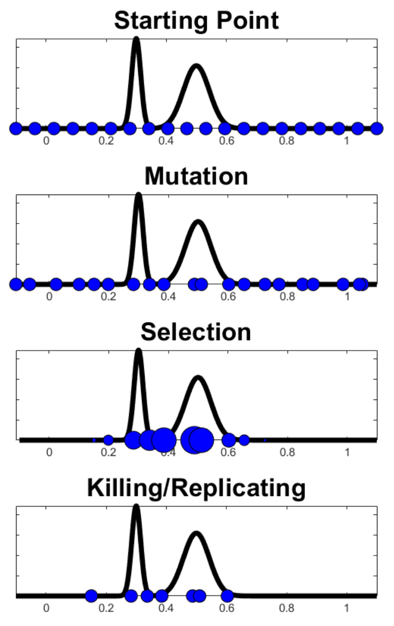
$$\begin{aligned}
\hat{\eta}_n(x_{1:n}) &= \frac{\hat{\gamma}_n(x_{1:n})}{Z_n} \\
&= \frac{p(x_{1:n}, y_{1:n})}{p(y_{1:n})} \\
&= p(x_{1:n} | y_{1:n})
\end{aligned}$$

Sequential Monte Carlo (SMC)

- **Aim:** Approximate $\hat{\eta}_n(x_n)$ using a set of N weighted particles $\{X_n^i, W_n^i\}_{i=1}^N$
- **Ingredients:**
  1. a Markov kernel to sample $X_n^i \sim M_n(\tilde{X}_{n-1}^i, \cdot)$
  2. a positive weight function to compute the weights $W_n^i \propto G_n(X_n^i)$
  3. a resampling scheme to obtain $\{\tilde{X}_n^i, 1/N\}_{i=1}^N$ from $\{X_n^i, W_n^i\}_{i=1}^N$

Example: the bootstrap particle filter corresponds to $\hat{\eta}_n(x_n) = p(x_n|y_{1:n})$, $M_n = f_n$ and $G_n = g(y_n|\cdot)$.

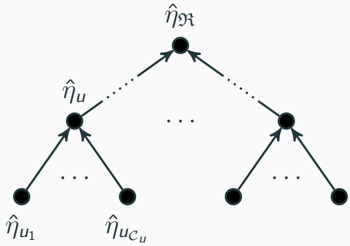
Sequential Monte Carlo (SMC)



Outline

Divide-and-Conquer SMC (DaC-SMC)

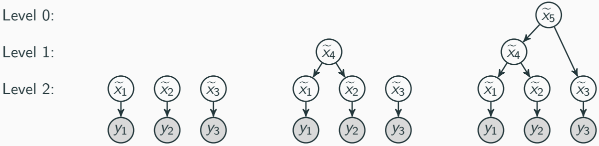
An extension of standard SMC in which the sequence of target distributions $\{\hat{\eta}_u\}_{u \in \mathbb{T}}$ evolves on a tree $\mathbb{T}$ rather than on a line (Lindsten et al., 2017).



For simplicity we will consider the unnormalised targets $\{\hat{\gamma}_u\}_{u \in \mathbb{T}}$, s.t. $\hat{\eta}_u = \hat{\gamma}_u / Z_u$.

Divide-and-Conquer SMC (DaC-SMC)

- easier to distribute/parallelise (Corenflos et al., 2022; Ding and Gandy, 2018)
- some models more naturally adapted to a tree structure than a ‘linear” one: e.g. graphical models (Lindsten et al., 2017; Paige and Wood, 2016; Jewell, 2015)



- distributed inference (Chan et al., 2021; Manderson and Goudie, 2021)

Divide-and-Conquer SMC (DaC-SMC)

At each node u we have a set of N weighted particles $\{X_u^i, W_u^i\}_{i=1}^N$ approximating $\hat{\eta}_u$.

To evolve use standard SMC ingredients

- transition/mutation: $X_u^i \sim M_u((X_{\ell(u)}^i, X_{r(u)}), \cdot)$
- reweighting: $W_u^i \propto G_u(X_u^i)$
- resampling

with the addition of a **merging** step when two branches of the tree merge.

The Merge Step

- **Aim:** Given two populations of weighted particles on the left and the right child, $\{X^i_{\ell(u)}, W^i_{\ell(u)}\}_{i=1}^N$ and $\{X^i_{r(u)}, W^i_{r(u)}\}_{i=1}^N$, approximating $\hat{\gamma}_{\ell(u)}$ and $\hat{\gamma}_{r(u)}$ build an approximation of $\hat{\gamma}_u$

The Merge Step

- **Aim:** Given two populations of weighted particles on the left and the right child, $\{X^i_{\ell(u)}, W^i_{\ell(u)}\}_{i=1}^N$ and $\{X^i_{r(u)}, W^i_{r(u)}\}_{i=1}^N$, approximating $\hat{\gamma}_{\ell(u)}$ and $\hat{\gamma}_{r(u)}$ build an approximation of $\hat{\gamma}_u$
- **Ingredients:**
  - the (weighted) product form estimator (Kuntz et al., 2022)

$$\hat{\gamma}^N_{\ell(u)} \times \hat{\gamma}^N_{r(u)} = \frac{1}{N^2} \sum_{i_1=1}^N \sum_{i_2=1}^N W^{i_1}_{\ell(u)} W^{i_2}_{r(u)} \delta_{(X^{i_1}_{\ell(u)}, X^{i_2}_{r(u)})}$$

  - N^2 *mixture* (importance) weights

$$m_u^{(i_1,i_2)} := W^{i_1}_{\ell(u)} W^{i_2}_{r(u)} \frac{\hat{\gamma}_u(x^{i_1}_{\ell(u)}, x^{i_2}_{r(u)})}{\hat{\gamma}_{\ell(u)}(x^{i_1}_{\ell(u)})\hat{\gamma}_{r(u)}(x^{i_2}_{r(u)})}$$

  - a resampling scheme to obtain $\{\tilde{X}^i_u, 1/N\}_{i=1}^N$ from $\{(X^{i_1}_{\ell(u)}, X^{i_2}_{r(u)}), m_u^{(i_2,i_2)}\}_{i_1,i_2=1}^N$

Cheaper Alternatives

- lightweight mixture resampling (Lindsten et al., 2017)
- lazy resampling schemes (Corenflos et al., 2022)
- strategies borrowed from the literature on incomplete U-statistics (Kuntz et al., 2023+)

Lightweight Mixture Resampling (Lindsten et al., 2017)

- Fix $\theta \ll N$.
- Sample $\{X_{\ell(u)}^{i_1}\}_{i_1=1}^{N\theta}$ from $\hat{\gamma}_{\ell(u)}$ and $\{X_{r(u)}^{i_2}\}_{i_2=1}^{N\theta}$ from $\hat{\gamma}_{r(u)}$.
- Build the $N\theta$ pairs $(X_{\ell(u)}^{i_1}, X_{r(u)}^{i_2})$ and compute their weights $m_u^{(i_1,i_2)}$.

Adaptive Lightweight Mixture Resampling

A strategy based on the effective sample size $\left(\sum_n m_u^n\right)^2 / \sum_n (m_u^n)^2$.

- Set a target ESS, $\mathrm{ESS}^\star$.
- For $i = 1, \dots, N$ build $(X^i_{\ell(u)}, X^i_{r(u)})$ and compute their weights $m_u^{(i,i)}$.
- While $\mathrm{ESS} < \mathrm{ESS}^\star$:
  - Draw one permutation of N, $\pi(N)$, build $(X^i_{\ell(u)}, X^{\pi(i)}_{r(u)})$ and compute their weights $m_u^{(i,\pi(i))}$.
  - Update ESS.

What resampling scheme?

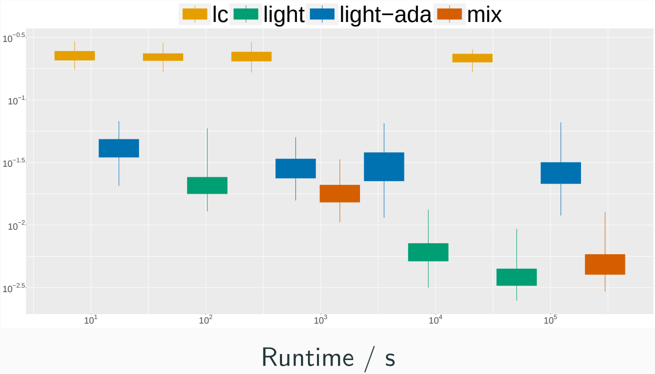


Figure 1: Average (over dimensions) MSE at $t = 10$ over 50 runs for $d = 128$ as function of runtime for the linear cost version of DaC in Lindsten et al. (2017) with lightweight mixture resampling, adaptive lightweight mixture resampling and full mixture resampling. The boxes, from left to right, correspond to increasing number of particles, $N = 100, 500, 1000, 5000$.

Theoretical Properties (Kuntz et al., 2023+)

- strong law of large numbers

$$\frac{1}{N}\sum_{i=1}^{N}\varphi(\tilde{X}_u^i) \overset{a.s.}{\to} \int \varphi(x)\hat{\eta}_u(x)\mathrm{d}x$$

- central limit theorem

$$\sqrt{N}\left(\frac{1}{N}\sum_{i=1}^{N}\varphi(\tilde{X}_u^i) - \int \varphi(x)\hat{\eta}_u(x)\mathrm{d}x\right) \overset{d}{\to} \mathcal{N}(0, \sigma_u^2(\varphi))$$

- unbiasedness of normalising constant estimates

And other: $\mathbb{L}^p$ errors, bias estimates, optimal intermediate targets, optimal proposals, . . .

Outline

State Space Models (SSM)

$$
\begin{tikzcd}
 & Y_{t-1} & Y_t & \\
\cdots \arrow[r] & X_{t-1} \arrow[u] \arrow[r, "f_t"] & X_t \arrow[u, "g_t"'] \arrow[r] & \cdots
\end{tikzcd}
$$

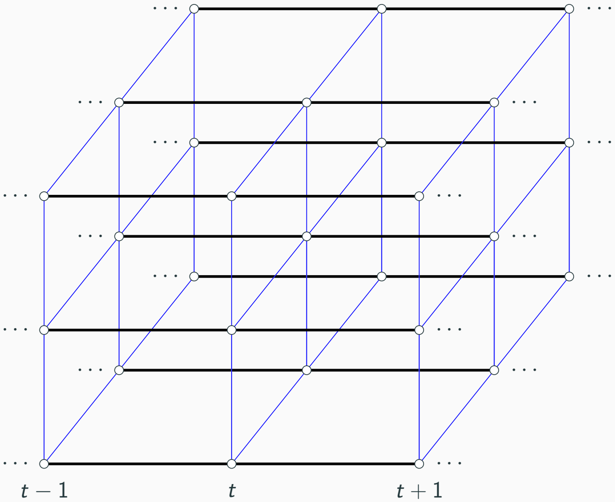
A state space model $(X_t, Y_t)_{t\geq 1} \subset \mathbb{R}^{d\times p}$ with

- transition density for the *latent* process $f_t(x_{t-1}, x_t)$
- likelihood for the *observation* process $g_t(x_t, y_t)$

We are interested in

$$p(x_{1:n}|y_{1:n}) \propto \hat{\gamma}_n(x_{1:n}) = p(x_{1:n-1}, y_{1:n-1}) f_n(x_{n-1}, x_n) g_n(x_n, y_n).$$

Spatial SSM



Collapse of Particle Filters

The mean squared error for a PF applied to a SSM with $(X_t)_{t\geq 1} \in \mathbb{R}^d$ is

$$\mathbb{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N}\varphi(\widetilde{X}_n^i) - \int p(x_n|y_{1:n})\varphi(x_n)\mathrm{d}x_n\right)^2\right] \leq \frac{C^d}{N}$$

For large d we need $N \propto e^d$ to get good results.

Particle Filters for High-Dimensions

Idea: Exploit the fact that dependencies in high dimensional SSMs encountered in practice are often local in space

- **Block PF (Rebeschini and Van Handel, 2015):** decompose of the state space into blocks and apply one step of a standard particle filter on each block, obtain an approximation of $p(x_n \mid y_{1:n})$ using the product of the approximations on each block

  Not asymptotically consistent (i.e. no converge to correct distribution)

Particle Filters for High-Dimensions

Idea: Exploit the fact that dependencies in high dimensional SSMs encountered in practice are often local in space

- **Space-Time PF (STPF; Beskos et al. (2017)):** decompose the space dimension into blocks and run independent particle filters on each of the blocks, then combine them using an importance resampling step.

  This requires explicit expression of the marginals $f_n(x_n(i) \mid x_n(i-1:1))$ and that the likelihood term factorizes so that $x_t(i)$ given the observations and the past only depends on a neighbourhood of $x_n(i)$, $\{x_n(j) : j \in \mathcal{A}\}$ for some $\mathcal{A} \subset \{1, \ldots, d\}$,

Particle Filters for High-Dimensions

Idea: Exploit the fact that dependencies in high dimensional SSMs encountered in practice are often local in space

- **Nested sequential Monte Carlo (NSMC; Næsseth et al. (2015)):** do one forward pass to approximate each dimension to include the dependence of the "previous" dimensions and one backward pass to introduce the dependence w.r.t. the "following" dimensions

  This requires explicit expression of the marginals $f_n(x_n(i) \mid x_n(i-1:1))$

Divide and Conquer SMC for Filtering

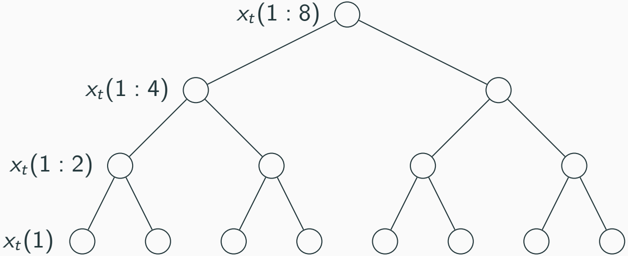


Figure 2: Space decomposition for $d = 8$.

Choice of Distributions

Fix n. We need $\{\hat{\gamma}_{n,u}\}_{u\in\mathbb{T}}$.

Constraints:

- $\hat{\gamma}_{n,\mathfrak{R}}(x_n) \propto p(x_n|y_{1:n})$
- for $u \neq \mathfrak{R}$, $\hat{\gamma}_{n,u}$ only depends on a subset of $x_n(1:d)$
- we can compute

$$\frac{\hat{\gamma}_{n,u}(x_{n,\ell(u)}, x_{n,r(u)})}{\hat{\gamma}_{n,\ell(u)}(x_{n,\ell(u)})\hat{\gamma}_{n,r(u)}(x_{n,r(u)})}$$

Choice of Distributions

Recall

$$p(x_n|y_{1:n}) \propto g_n(x_n, y_n) \int f_n(x_{n-1}, x_n) p(x_{n-1}|y_{1:n-1}) \mathrm{d}x_{n-1}.$$

Use auxiliary functions $f_{n,u}, g_{n,u}$ such that $f_{n,\mathfrak{R}} = f_n$, $g_{n,\mathfrak{R}} = g_n$ and for $u \in \mathbb{T} \setminus \mathfrak{R}$, $f_{n,u}$ and $g_{n,u}$ serve as proxies for marginals of f_n, g_n

$$\hat{\gamma}_{n,u}(x_{n,u}) = g_{n,u}(x_{n,u}, (y_n(i))_{i\in u}) \int f_{n,u}(x_{n-1}, x_{n,u}) \hat{\gamma}_{n-1,\mathfrak{R}}(x_{n-1}) \mathrm{d}x_{n-1}$$

Choice of Distributions

Recall

$$p(x_n|y_{1:n}) \propto g_n(x_n, y_n) \int f_n(x_{n-1}, x_n) p(x_{n-1}|y_{1:n-1}) \mathrm{d}x_{n-1}.$$

Use auxiliary functions $f_{n,u}, g_{n,u}$ such that $f_{n,\mathfrak{R}} = f_n$, $g_{n,\mathfrak{R}} = g_n$ and for $u \in \mathbb{T} \setminus \mathfrak{R}$, $f_{n,u}$ and $g_{n,u}$ serve as proxies for marginals of f_n, g_n

$$\hat{\gamma}_{n,u}(x_{n,u}) = g_{n,u}(x_{n,u}, (y_n(i))_{i \in u}) \int f_{n,u}(x_{n-1}, x_{n,u}) \hat{\gamma}_{n-1,\mathfrak{R}}(x_{n-1}) \mathrm{d}x_{n-1}$$

The integral above is intractable, so we define approximate targets as in marginal particle filters (Klaas et al., 2005)

$$\hat{\gamma}_{n,u}(x_{n,u}) = g_{n,u}(x_{n,u}, (y_n(i))_{i \in u}) W_{t-1}^i \sum_{i=1}^{N} f_{n,u}(X_{n-1,\mathfrak{R}}^i, x_{n,u})$$

Divide and Conquer SMC for Filtering

At time t:

- start with $\{X^i_{n-1,\mathfrak{R}}, W^i_{n-1}\}_{i=1}^N$ approximating $p(x_{n-1}|y_{1:n-1})$ at the leaf level
- define targets

$$\hat{\gamma}_{n,u}(x_{n,u}) = g_{n,u}(x_{n,u}, (y_n(i))_{i\in u}) W^i_{t-1} \sum_{i=1}^N f_{n,u}(X^i_{n-1,\mathfrak{R}}, x_{n,u})$$

- use DaC to move up the space from the leaves to the root, i.e. from d particle populations approximating 1-dimensional to one d-dimensional population $\{X^i_{n,\mathfrak{R}}, W^i_n\}_{i=1}^N$ which approximates $p(x_n|y_{1:n})$

Pros and Cons

Pros

- No need for analytical form of $f_n(x_n(i) \mid x_n(1:i-1))$
- No need for factorised likelihoods
- Easy to parallelise and distribute

Cons

- Polynomial cost in N (can be mitigated via GPUs)
- Needs specification of $\hat{\gamma}_{n,u}$

Linear Gaussian Model

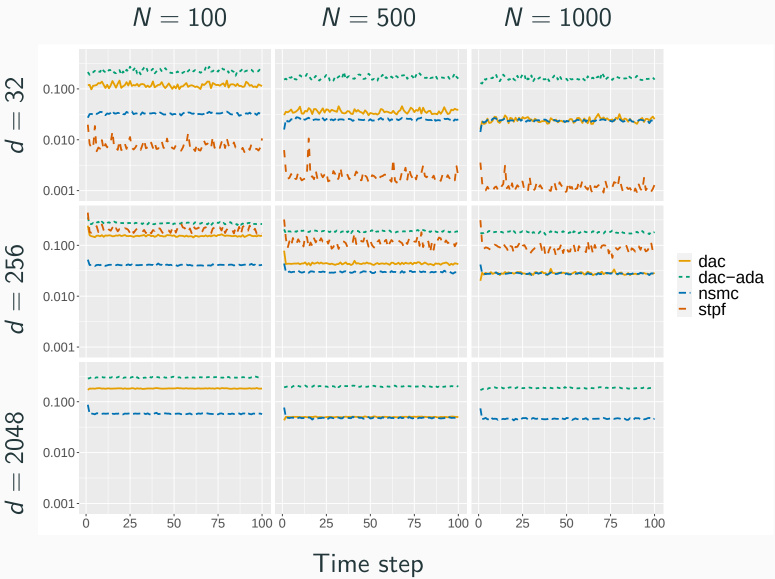
$$f_1(x_1) = \mathcal{N}(x_1; m_1, \Sigma_1)$$
$$f_t(x_{t-1}, x_t) = \mathcal{N}(x_t; Ax_{t-1}, \Sigma)$$
$$g_t(x_t, y_t) = \mathcal{N}(y_t; x_t, \sigma_y^2 Id_d),$$

with $\Sigma \in \mathbb{R}^{d \times d}$ a tridiagonal matrix.

We compare the RMSE

$$\text{RMSE}(x_t(i)) := \frac{\mathbb{E}\left[(\bar{x}_t(i) - \mu_{t,i})^2\right]}{\sigma_{t,i}^2}$$

Linear Gaussian Model - RMSE



Spatial Model

The components of X_t are indexed by the vertices $v \in V$ of a lattice, where $V = \{1, \ldots, d\}^2$, and

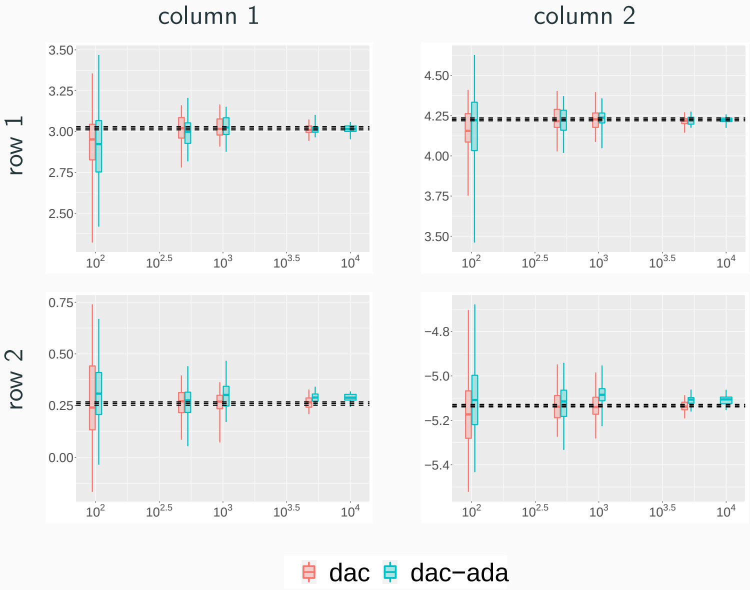
$$X_t(v) = X_{t-1}(v) + U_t(v), \qquad U_t(v) \sim \mathcal{N}(0, \sigma_x^2)$$

and

$$g_t(x_t, y_t) \propto$$

$$\left[1 + \nu^{-1} \sum_{v \in V} \left((y_t(v) - x_t(v)) \sum_{j : D(v,j) \leq r_y} \tau^{D(v,j)} (y_t(j) - x_t(j)) \right) \right]^{-(\nu + d^2)/2}.$$

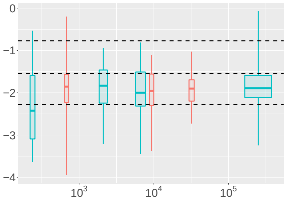
2 × 2 grid



column 1
column 2
row 1
row 2
3.50
3.25
3.00
2.75
2.50
4.50
4.25
4.00
3.75
3.50
0.75
0.50
0.25
0.00
−4.8
−5.0
−5.2
−5.4
10^2
$10^{2.5}$
10^3
$10^{3.5}$
10^4
dac
dac−ada

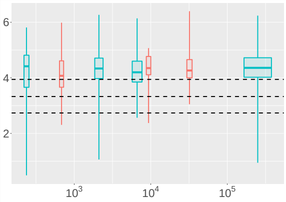
8×8 **grid**

$(1, 1)$



Runtime / s

$(8, 6)$



Runtime / s

dac dac−ada

Multivariate Stochastic Volatility Model

$$X_1 \sim \mathcal{N}_d(0, \Sigma_0),$$
$$Y_t = C_t^{1/2} V_t,$$
$$X_{t+1} = \Phi X_t + U_t,$$

with C_t a diagonal matrix with entries

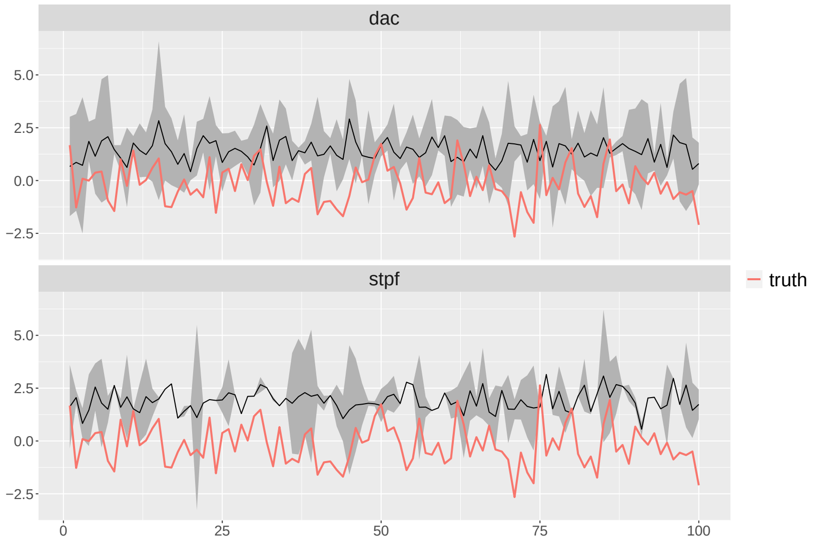
$$C_t = \mathrm{diag}\left(\exp(X_t(1)), \exp(X_t(2)), \ldots, \exp(X_t(d))\right),$$

Φ a diagonal matrix with diagonal entries ϕ_i,

$$\begin{pmatrix} V_t \\ U_t \end{pmatrix} \sim \mathcal{N}_{2d}(0, \Sigma) \qquad \text{with } \Sigma = \begin{pmatrix} \Sigma_{VV} & \Sigma_{VU} \\ \Sigma_{VU} & \Sigma_{UU} \end{pmatrix},$$

and Σ_0 has entries given by

$$(\Sigma_0)_{ij} = (\Sigma_{UU})_{ij} / (1 - \phi_i \phi_j),$$


dac
stpf
5.0
2.5
0.0
−2.5
0
25
50
75
100
truth

Opportunities and Related Ideas

- Theoretical guarantees should follow from those of standard DaC-SMC (Kuntz et al., 2023+) and marginal particle filters (Crucinio and Johansen, 2023)
- smoothing, parameter estimation
- Other applications: Parallel (in time) Smoothing (Corenflos et al., 2022; Ding and Gandy, 2018), Divide and Conquer Fusion (Chan et al., 2021), inference for graphical models and phylogenetic trees (Jewell, 2015; Paige and Wood, 2016; Lindsten et al., 2017)

Thank you!

Bibliography i

Alexandros Beskos, Dan Crisan, Ajay Jasra, Kengo Kamatani, and Yan Zhou. A stable particle filter for a class of high-dimensional state-space models. *Advances in Applied Probability*, 49(1):24–48, 2017.

Frédéric Cérou, Pierre Del Moral, Teddy Furon, and Arnaud Guyader. Sequential monte carlo for rare event estimation. *Statistics and computing*, 22(3):795–808, 2012.

R. Chan, M. Pollock, A. M. Johansen, and G. O. Roberts. Divide-and-conquer Monte Carlo fusion. *arXiv preprint arXiv:2110.07265*, 2021.

A. Corenflos, N. Chopin, and S. Särkkä. De-Sequentialized Monte Carlo: a parallel-in-time particle smoother. *Journal of Machine Learning Research*, 23(283):1–39, 2022.

F. R. Crucinio and A. M. Johansen. Properties of marginal sequential Monte Carlo methods. *arXiv preprint arXiv:2303.03498*, 2023.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, 22:1009–1020, 2012.

D. Ding and A. Gandy. Tree-based particle smoothing algorithms in a hidden markov model. *arXiv preprint arXiv:1808.08400*, 2018.

S. W. Jewell. *Divide and conquer sequential Monte Carlo for phylogenetics*. Master thesis, University of British Columbia, 2015.

Mike Klaas, Nando De Freitas, and Arnaud Doucet. Toward practical N^2 Monte Carlo: The marginal particle filter. In *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence*, pages 308–315, 2005.

J. Kuntz, F. R. Crucinio, and A. M. Johansen. The divide-and-conquer sequential Monte Carlo algorithm: theoretical properties and limit theorems. *Annals of Applied Probability*, 2023+.

Juan Kuntz, Francesca R Crucinio, and Adam M Johansen. Product-form estimators: exploiting independence to scale up Monte Carlo. *Statistics and Computing*, 32(1):1–22, 2022.

Fredrik Lindsten, Adam M Johansen, Christian A Næsseth, Bonnie Kirkpatrick, Thomas B Schön, John A D Aston, and Alexandre Bouchard-Côté. Divide-and-Conquer with sequential Monte Carlo. *Journal of Computational and Graphical Statistics*, 26(2):445–458, 2017.

Andrew A Manderson and Robert JB Goudie. Combining chains of bayesian models with markov melding. *arXiv preprint arXiv:2111.11566*, 2021.

Bibliography ii

Christian A Næsseth, Fredrik Lindsten, and Thomas B Schön. Nested sequential Monte Carlo methods. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1292–1301. Proceedings of Machine Learning Research, 2015.

B. Paige and F. Wood. Inference networks for sequential monte carlo in graphical models. In *33rd Int Conf Mach Learn*, volume 48, pages 3040–3049, 2016.

Patrick Rebeschini and Ramon Van Handel. Can local particle filters beat the curse of dimensionality? *Annals of Applied Probability*, 25 (5):2809–2866, 2015.

S. A. Sisson, Y. Fan, and M. M. Tanaka. Sequential Monte Carlo without likelihoods. 104(4):1760–1765, February 2007.