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

Francesca R. Crucinio, King's College London

ISBA Satellite workshop

Joint work with Adam M. Johansen

Outline

Spatio-temporal Data

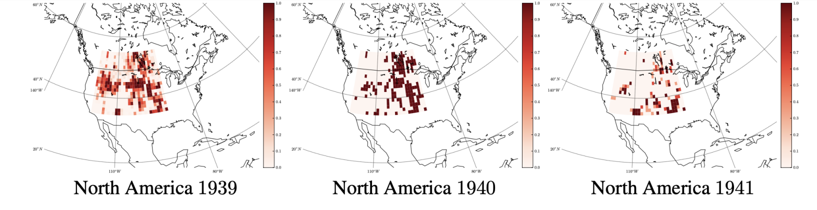


Figure 1: Drought Detection from Rain Precipitations (Næsseth et al., 2015).

- **Observe** precipitation (in millimeters) for each location and year
- **Recover** if there is a drought in that location

State Space Models (SSM)

$$
\begin{tikzcd}
 & Y_{n-1} & Y_n & \\
\cdots \arrow[r] & X_{n-1} \arrow[u] \arrow[r, "f_n"] & X_n \arrow[u, "g_n"'] \arrow[r] & \cdots
\end{tikzcd}
$$

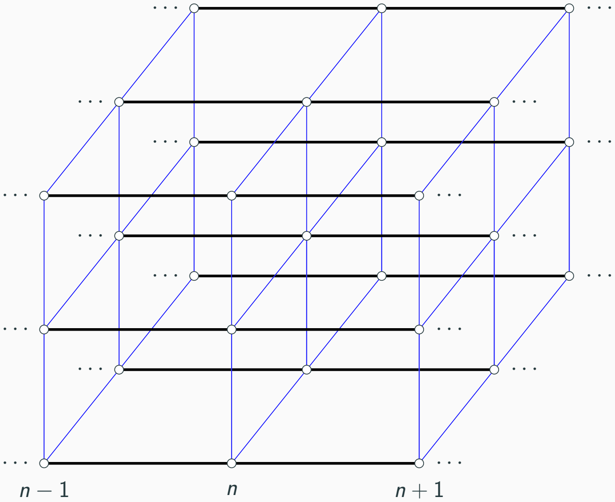
A state space model $(X_n, Y_n)_{n\geq 1} \subset \mathbb{R}^{d\times p}$ with

- transition density for the *latent* process $f_n(x_{n-1}, x_n)$
- likelihood for the *observation* process $g_n(x_n, y_n)$

We are interested in

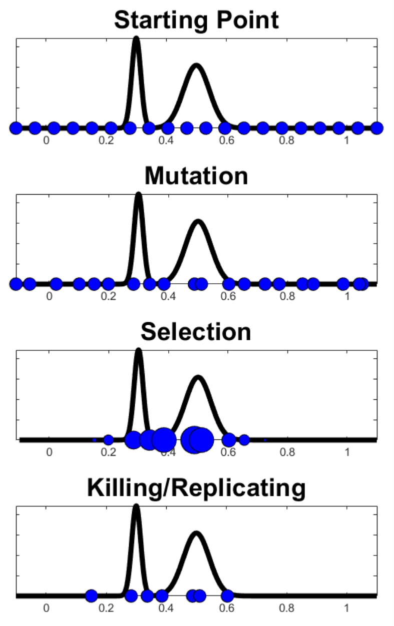
$$p(x_{1:n}|y_{1:n}) \propto p(x_{1:n-1}, y_{1:n-1})f_n(x_{n-1}, x_n)g_n(x_n, y_n).$$

Spatial State Space Model



Outline

Sequential Monte Carlo (SMC)



Sequential Monte Carlo (SMC)

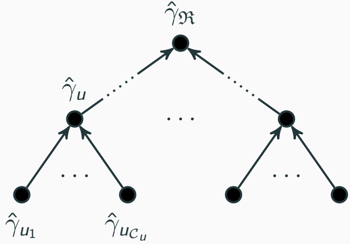
- **Aim:** Approximate $p(x_n|y_{1:n})$ using a set of N weighted particles $\{X_n^i, W_n^i\}_{i=1}^N$
- **Ingredients:**
  1. a transition density to sample $X_n^i \sim f_n(\tilde{X}_{n-1}^i, \cdot)$
  2. a likelihood function to compute the weights $W_n^i \propto g_n(X_n^i)$
  3. a resampling scheme to obtain $\{\tilde{X}_n^i, 1/N\}_{i=1}^N$ from $\{X_n^i, W_n^i\}_{i=1}^N$

Also known as **bootstrap particle filter**.

Outline

Divide-and-Conquer SMC (DaC-SMC)

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



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 $\hat{\gamma}_u$.

To evolve use standard SMC ingredients

- transition/mutation: $X_u^i \sim M_u((X_{\ell(u)}^i, X_{r(u)}^i), \cdot)$
- 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 $\{(X^{i_1}_{\ell(u)}, X^{i_2}_{r(u)}), m_u^{(i_2,i_2)}\}_{i_1,i_2=1}^N$

Outline

Divide and Conquer SMC for Filtering

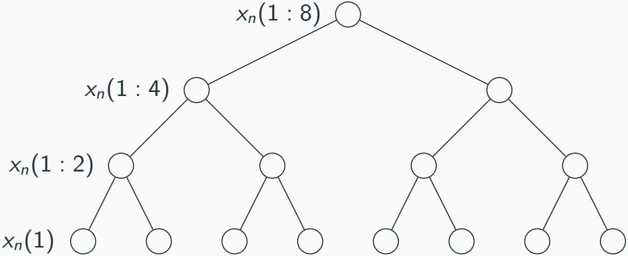


Figure 2: Space decomposition for $d = 8$.

Divide and Conquer SMC for Filtering

At time n:

- 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

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

- 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 $p(x_n|y_{1:n})$

Linear Gaussian Model

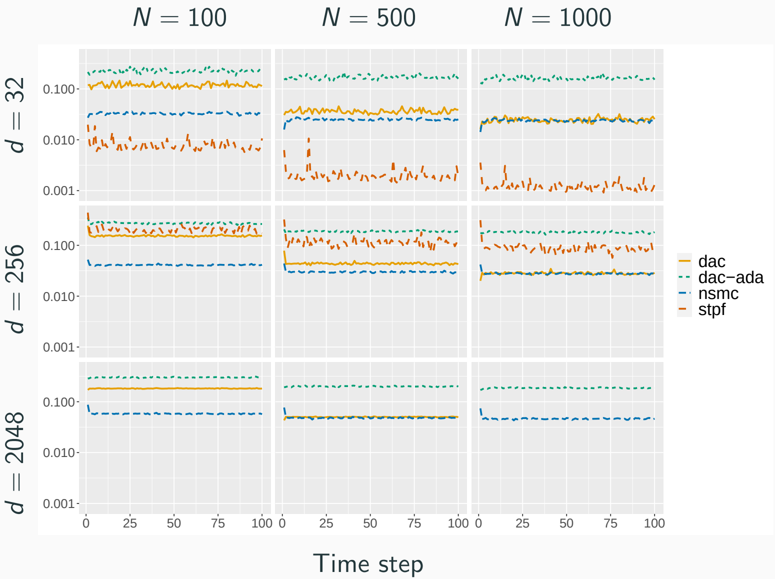
$$\begin{aligned} f_1(x_1) &= \mathcal{N}(x_1; m_1, \Sigma_1) \\ f_n(x_{n-1}, x_n) &= \mathcal{N}(x_n; Ax_{n-1}, \Sigma) \\ g_n(x_n, y_n) &= \mathcal{N}(y_n; x_n, \sigma_y^2 Id_d), \end{aligned}$$

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

We compare the RMSE

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

Linear Gaussian Model - RMSE



Spatial Model

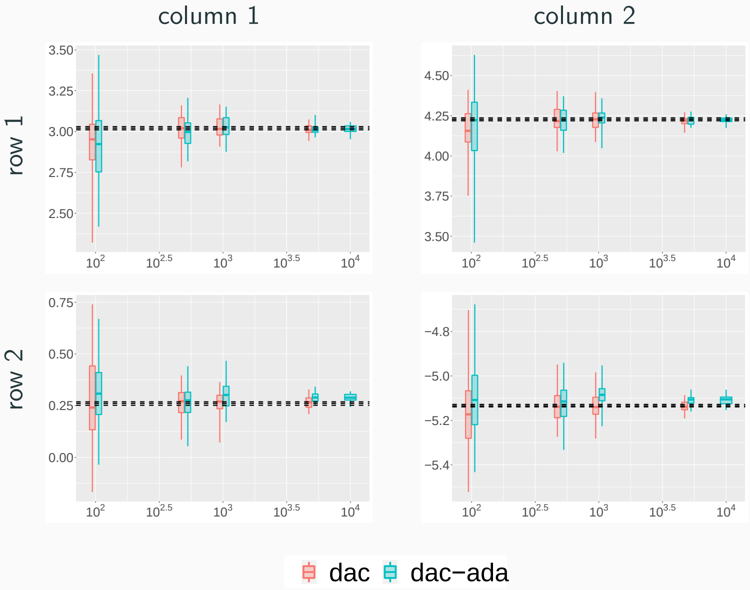
The components of X_t are indexed by the vertices $v \in V$ of a lattice, where $V = \{1, \dots, d\}^2$, and

$$X_n(v) = X_{n-1}(v) + U_n(v), \qquad U_n(v) \sim \mathcal{N}(0, \sigma_x^2)$$

and

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

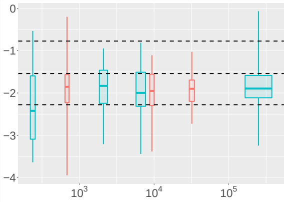
2 × 2 grid



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

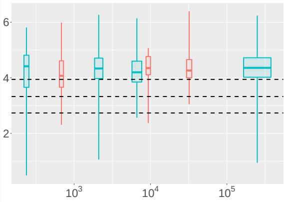
8×8 grid

$(1, 1)$



Runtime / s

$(8, 6)$



Runtime / s

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 $\hat{\gamma}_{n,u}$

Thank you!

Bibliography i

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.

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.

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.