Divide-and-Conquer sequential Monte Carlo for high dimensional filtering

Francesca R. Crucinio
Joint work with Adam M. Johansen

6 February 2023

CREST-CMAP Statistics Seminar

Outline

Spatio-temporal Data

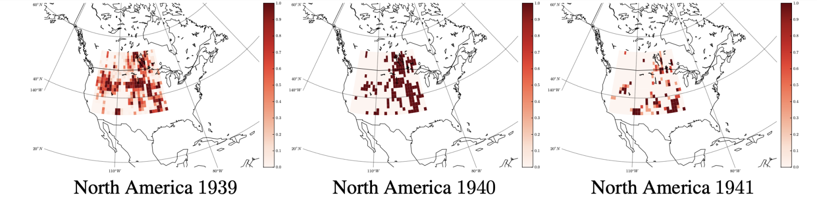


Figure 1: Drought Detection from Rain Precipitations (Næsseth et al., 2015).

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

Spatio-temporal Data

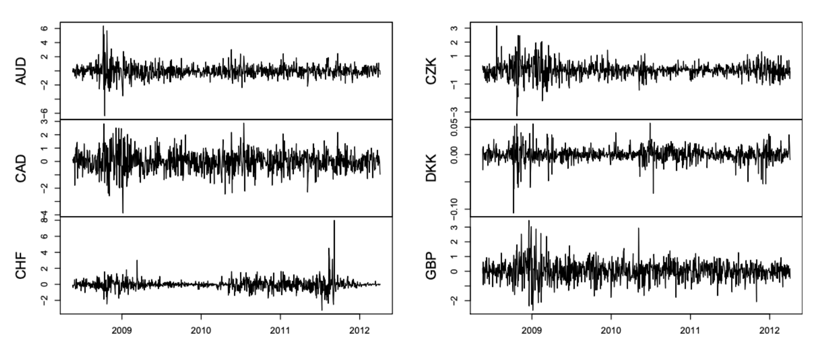


Figure 2: Percentage log returns of six EUR exchange rates (Hosszejni and Kastner, 2021).

- **Observe** exchange rate w.r.t. EUR
- **Recover** factors driving volatility

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 $g_t(x_t, y_t)$

Inference for State Space Models

We want to estimate the *filtering* distribution

$$p(x_{1:t} \mid y_{1:t}) \propto p(x_{1:t}, y_{1:t}) = \prod_{k=1}^{t} f_k(x_{k-1}, x_k) g_k(x_k, y_k),$$

with the convention that $f_1(x_0, x_1) \equiv f_1(x_1)$.

- MCMC
- Particle filters (PF)

Why PF? PF proceed iteratively exploiting the fact that

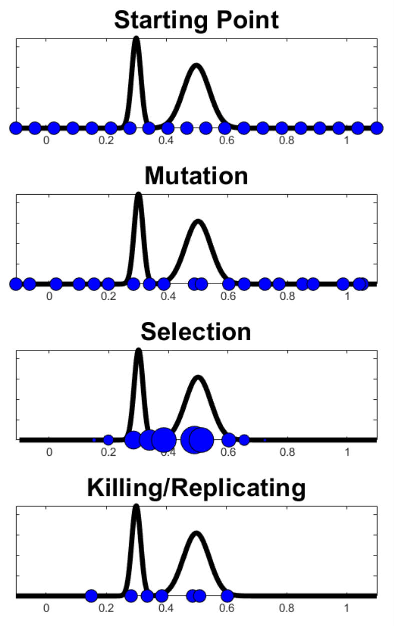
$$\begin{aligned} p(x_{1:t}, y_{1:t}) &= \prod_{k=1}^{t} f_k(x_{k-1}, x_k) g_k(x_k, y_k) \\ &= p(x_{1:t-1}, y_{1:t-1}) f_t(x_{t-1}, x_t) g_k(x_t, y_t) \end{aligned}$$

Particle Filters (PF)

- **Aim:** Approximate $p(x_t|y_{1:t})$ using a set of N weighted particles $\{X_t^i, W_t^i\}_{n=1}^N$
- **Ingredients:**
    1. a resampling scheme to obtain $\{\tilde{X}_{t-1}^i, 1/N\}_{n=1}^N$ from $\{X_{t-1}^i, W_{t-1}^i\}_{n=1}^N$
    2. the transition density to sample $X_t^i \sim f_t(\tilde{X}_{t-1}^i, \cdot)$
    3. the likelihood to compute the weights $W_t^i \propto g_t(X_t^i, Y_t^i)$

This is the bootstrap particle filter, more sophisticated filters also exist!

Particle Filters



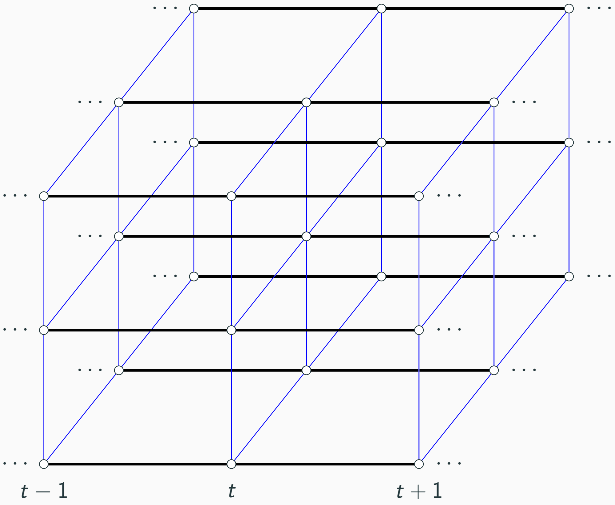
Collapse of Particle Filters

The mean squared error for a PF applied to a SSM with $X_t \in \mathbb{R}^d$ is

$$\mathbb{E}\left[\left(\sum_{i=1}^{N} W_t^i \varphi(X_t^i) - \int p(x_t|y_{1:t})\varphi(x_t)dx_t\right)^2\right] \leq \frac{C^d}{N}$$

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

Spatial SSM



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 on each block, obtain an approximation to $p(x_t \mid y_{1:t})$ 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 N independent particle filters on each of the blocks, then combine them using an importance resampling step.

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

Outline

Divide and Conquer SMC for Filtering

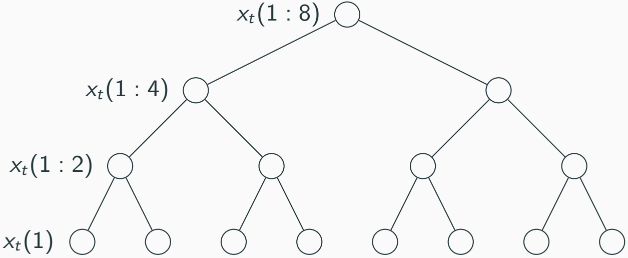


Figure 3: Space decomposition for $d = 8$.

Divide and Conquer SMC

An extension of standard SMC (Chopin and Papaspiliopoulos, 2020) in which the sequence of target distributions $\{\gamma_u\}_{u\in\mathbb{T}}$ evolves on a tree $\mathbb{T}$ rather than on a line (Lindsten et al., 2017).

Need to define a sequence of distributions $\{\gamma_u\}_{u\in\mathbb{T}}$ so that at the root of the tree $\mathfrak{R}$ we have

$$\gamma_{\mathfrak{R}}(x_t(1:d)) = p(x_t|y_{1:t})$$

and for all other nodes γ_u only depends on a subset of $x_t(1:d)$, e.g. at the leaf level we have d distributions each depending on one component of x_t, $\gamma_u(x_t(i))$ for $i = 1, \ldots, d$.

Divide and Conquer SMC

Uses standard SMC ingredients

- transition/mutation
- reweighting
- resampling

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

But instead of moving only “forward in time” we move “forward in space”

The Merge Step

Start with a population 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 $\gamma_{\ell(u)}$ and $\gamma_{r(u)}$.

Want to approximate γ_u using $\{X^i_{\ell(u)}, W^i_{\ell(u)}\}_{i=1}^N$ and $\{X^i_{r(u)}, W^i_{r(u)}\}_{i=1}^N$.

Use **importance sampling**!

The Merge Step

- Build the (weighted) product form estimator (Kuntz et al., 2022)

$$\gamma_{\ell(u)}^N \times \gamma_{r(u)}^N = \frac{1}{N^2} \sum_{i_1=1}^{N} \sum_{i_2=1}^{N} W_{\ell(u)}^{i_1} W_{r(u)}^{i_2} \delta_{(X_{\ell(u)}^{i_1}, X_{r(u)}^{i_2})}$$

- Reweight using *mixture* (importance) weights

$$m_u(x_{\ell(u)}, x_{r(u)}) := \frac{\gamma_u(x_{\ell(u)}, x_{r(u)})}{\gamma_{\ell(u)}(x_{\ell(u)})\gamma_{r(u)}(x_{r(u)})}$$

  capture the mismatch between γ_u and $\gamma_{\ell(u)} \times \gamma_{r(u)}$
- Compute the weights $\widetilde{W}_u^{i_1,i_2} = W_{\ell(u)}^{i_1} W_{r(u)}^{i_2} m_u(X_{\ell(u)}^{i_1}, X_{r(u)}^{i_2})$
- Resample a new particle population of size N and set $W_u^i = 1/N$ for $i = 1, \ldots, N$

Divide and Conquer SMC

- merge the particle populations on u's children and obtain $\gamma_{\ell(u)}^N \times \gamma_{r(u)}^N$
- reweight the particles using m_u
- resample to get N particles with equal weights
- (optional) mutate the particles using a Markov kernel K_u

We move from d 1-dimensional families of particles to one d-dimensional family which approximates $p(x_t|y_{1:t})$

Choice of Distributions

Fix t. We need $\{\gamma_{t,u}\}_{u \in \mathbb{T}}$.

Constraints:

- $\gamma_{t,\mathfrak{R}}(x_t) = p(x_t | y_{1:t})$
- for $u \neq \mathfrak{R}$, $\gamma_{t,u}$ only depends on a subset of $x_t(1:d)$
- we can compute

$$m_{t,u}(x_{t,\ell(u)}, x_{t,r(u)}) = \frac{\gamma_{t,u}(x_{t,\ell(u)}, x_{t,r(u)})}{\gamma_{t,\ell(u)}(x_{t,\ell(u)})\gamma_{t,r(u)}(x_{t,r(u)})}$$

Choice of Distributions

Recall

$$p(x_t|y_{1:t}) = g_t(x_t, y_t) \int f(x_{t-1}, x_t) p(x_{t-1}|y_{1:t-1}) \mathrm{d}x_{t-1}.$$

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

$$\gamma_{t,u}(x_{t,u}) = g_{t,u}(x_{t,u}, (y_t(i))_{i \in u}) \int f_{t,u}(x_{t-1}, x_{t,u}) \gamma_{t-1,\mathfrak{R}}(x_{t-1}) \mathrm{d}x_{t-1}$$

Marginal SMC

Problem: The integral in

$$\gamma_{t,u}(x_{t,u}) = g_{t,u}(x_{t,u}, (y_t(i))_{i\in u}) \int f_{t,u}(x_{t-1}, x_{t,u})\gamma_{t-1,\mathfrak{R}}(x_{t-1})\mathrm{d}x_{t-1}$$

is intractable

Solution: Define approximate targets as in marginal particle filters (Klaas et al., 2005)

$$\gamma_{t,u}(x_{t,u}) = g_{t,u}(x_{t,u}, (y_t(i))_{i\in u})\frac{1}{N}\sum_{i=1}^{N} f_{t,u}(x^i_{t-1,\mathfrak{R}}, x_{t,u})$$

Divide and Conquer SMC for Filtering

At time t:

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

$$\gamma_{t,u}(x_{t,u}) = g_{t,u}(x_{t,u}, (y_t(i))_{i\in u})\frac{1}{N}\sum_{i=1}^N f_{t,u}(x^i_{t-1,\mathfrak{R}}, x_{t,u})$$

- use DaC to move up the space from the leaves to the root
- at the root obtain $\{x^i_{t,\mathfrak{R}}, 1/N\}_{i=1}^N$ approximating $p(x_t|y_{1:t})$

Pros and Cons

Pros

- No need for analytical form of $p(x_t(i) \mid x_t(i-1:1))$
- No need for factorized likelihoods
- Easy to parallelize and distribute

Cons

- Polynomial cost in N (can be mitigated via GPUs)
- Needs specification of γ_u

1. Motivation

2. State Space Models and Particle Filters

3. Divide and Conquer SMC for Filtering
   - Main Ideas
   - Experiments

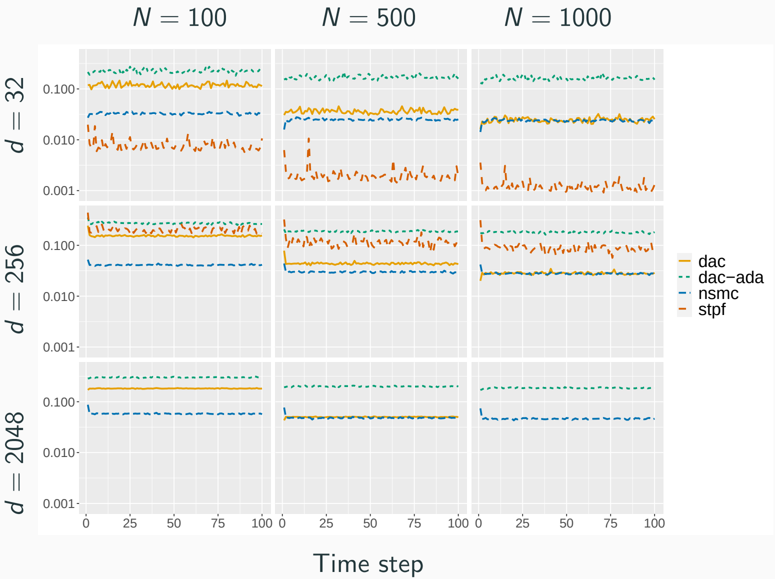
Linear Gaussian Model

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

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

For this model the distribution $p(x_t | y_{1:t})$ can be obtained in closed form using the **Kalman filter** and amounts to matrix inversion/multiplication.

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

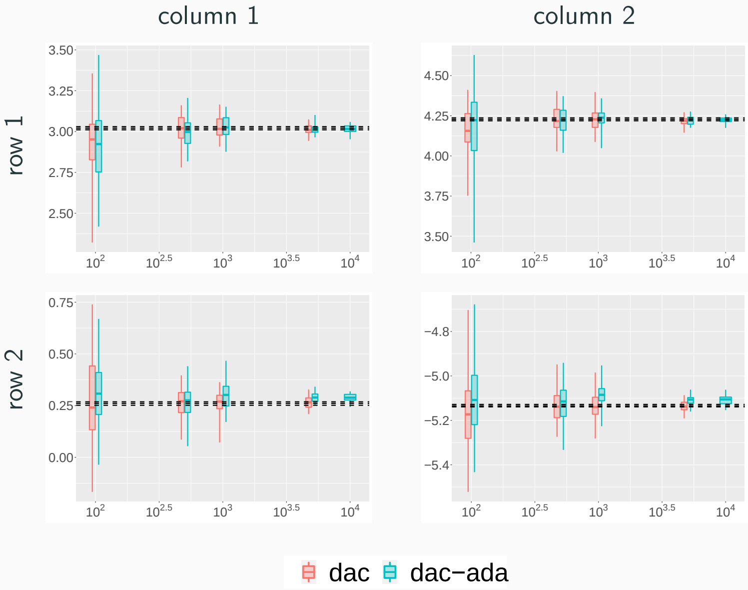
$$X_t(v) = X_{t-1}(v) + U_t(v), \qquad U_t(v) \sim \mathcal{N}(0, \sigma_x^2)$$

and

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

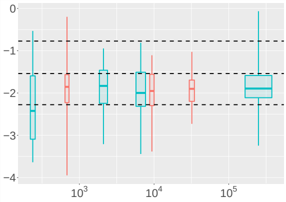
$$\left[1 + \nu^{-1} \sum_{v \in V} \left((y_t(v) - x_t(v)) \sum_{j : D(v,j) \le r_y} \tau^{D(v,j)} (y_t(j) - x_t(j)) \right) \right]^{-(\nu + d^2)/2} .$$

2×2 grid



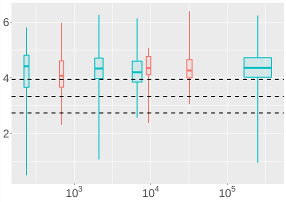
8×8 grid

$(1, 1)$



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

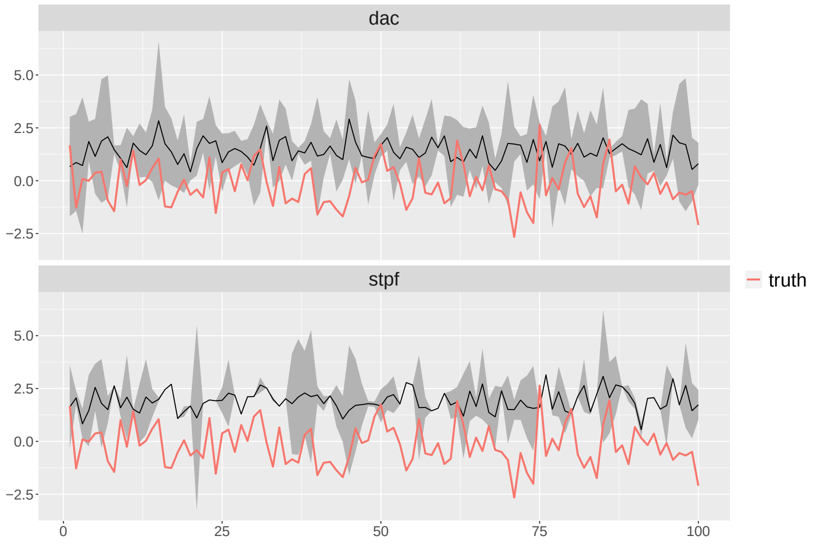
$$C_t = \mathrm{diag}\left(\exp(X_t(1)), \exp(X_t(2)), \dots, \exp(X_t(d))\right),$$

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

What next?

- Theoretical guarantees should follow from those of standard DaC-SMC (Kuntz et al., 2021)
- Test on real models/data

- smoothing, parameter estimation, ...
- lookahead methods

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.

N Chopin and O Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer, Cham, 2020. doi: 10.1007/978-3-030-47845-2.

Darjus Hosszejni and Gregor Kastner. Modeling univariate and Multivariate Stochastic Volatility in R with stochvol and factorstochvol. *Journal of Statistical Software*, 100:1–34, 2021.

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. *arXiv preprint arXiv:2110.15782*, 2021.

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.

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