Sequential Monte Carlo methods for probabilistic graphical models

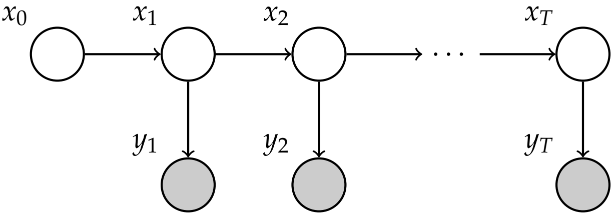
Christian A. Naesseth
Division of Automatic Control
Department of Electrical Engineering
Linköping University, Sweden

Ongoing joint work with: **Fredrik Lindsten** (University of Cambridge), **Thomas Schön** (Uppsala University), **Adam M. Johansen** (University of Warwick), **Bonnie Kirkpatrick** (University of Miami), **John Aston** (University of Cambridge) and **Alexandre Bouchard-Côté** (University of British Columbia).

Background – graphical models (I/II)

A **probabilistic graphical model** (PGM) is a probabilistic model where a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ represents the conditional independency structure between random variables,

1. a set of **vertices** $\mathcal{V}$ (nodes) represents the random variables
2. a set of **edges** $\mathcal{E}$ containing elements $(i, j) \in \mathcal{E}$ connecting a pair of nodes $(i, j) \in \mathcal{V} \times \mathcal{V}$



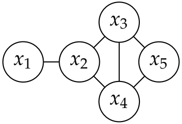
$$p(x_{0:T}, y_{1:T}) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1}) \prod_{t=1}^{T} p(y_t \mid x_t).$$

Background – graphical models (II/II)

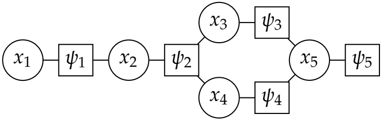
For an undirected graphical model (Markov random field), the joint PDF over all the involved random variables is

$$p(X_{\mathcal{V}}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(X_C),$$

where $\mathcal{C}$ is the set of cliques in $\mathcal{G}$, and $Z = \int \prod_{C \in \mathcal{C}} \psi_C(X_C) dX_{\mathcal{V}}$.



Undirected graph



Example of a **factor graph** making interactions explicit, $p(x_{1:5}) = \frac{1}{Z} \prod_{i=1}^{5} \psi_i(\cdot)$.

Background – sequential Monte Carlo (I/II)

Approximate a **sequence** of probability distributions on a sequence of probability spaces of **increasing dimension**.

Let $\{\gamma_k(x_{1:k})\}_{k\geq 1}$ be a sequence of unnormalised densities and

$$\bar{\gamma}_k(x_{1:k}) = \frac{\gamma_k(x_{1:k})}{Z_k}$$

Approximates

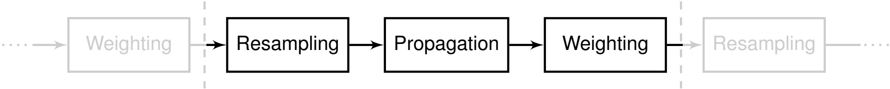
$$\bar{\gamma}_k(x_{1:k}) \approx \sum_{i=1}^{N} \frac{w_k^i}{\sum_{l=1}^{N} w_k^l} \delta_{x_{1:k}^i}(x_{1:k}).$$

Ex. (SSM)

$$\bar{\gamma}_k(x_{1:k}) = p(x_{1:k} \mid y_{1:k}), \qquad \gamma_k(x_{1:k}) = p(x_{1:k}, y_{1:k}),$$

$$Z_k = p(y_{1:k}).$$

Background – sequential Monte Carlo (II/II)



SMC = resampling + sequential importance sampling

Given, $\{x^i_{1:k-1}, w^i_{k-1}\}_{i=1}^N$, repeat for $i = 1, \dots, N$:

1. **Resampling:** $\mathbb{P}(\check{x}^i_{1:k-1} = x^j_{1:k-1}) = w^j_{k-1} / \sum_l w^l_{k-1}$.
2. **Propagation:** $x^i_k \sim r_k(x_k \mid \check{x}^i_{1:k-1})$ and $x^i_{1:k} = \{\check{x}^i_{1:k-1}, x^i_k\}$.
3. **Weighting:** $w^i_k = W_k(x^i_{1:k}) = \frac{\gamma_k(x^i_{1:k})}{\gamma_{k-1}(x^i_{1:k-1}) r_k(x^i_k | x^i_{1:k-1})}$.

(a hopefully) intuitive preview

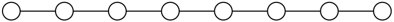
SMC samplers are used to approximate a sequence of probability distributions on a sequence of probability spaces.

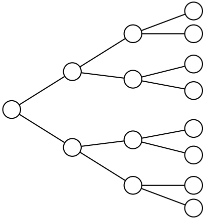
Using an artificial sequence of intermediate target distributions for an SMC sampler is a powerful (and **quite possibly underutilised**) idea.

Key idea: Perform and make use of **decompositions** of graphical models to design SMC inference methods.

Outline

1. Example – from information theory
2. Sequential decomposition → “standard” SMC
    a) Sequential decomposition and SMC for PGMs
    b) Examples – Estimating partition functions
3. Particle Markov chain Monte Carlo and partial blocking
    a) Particle Gibbs
    b) Partial blocking
    c) Example – Gaussian Markov random field
4. Conclusions





Information theory – 2D channel capacity

Example borrowed from:

M. Molkaraie and H.-A. Loeliger, **Monte Carlo algorithms for the partition function and information rates of two-dimensional channels**, *IEEE Transactions on Information Theory*, 59(1): 495–503, 2013.

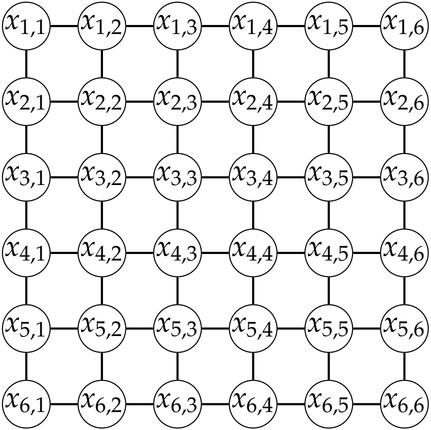
2D binary-input channel with the **constraint** that no two horizontally or vertically adjacent variables may be both be equal to 1.

$$\begin{array}{ccccc} \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 0 & 1 & 0 & \cdots \\ \cdots & 0 & 0 & 1 & \cdots \\ \cdots & 0 & 1 & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{array}$$

Of interest in emerging magnetic and optical storage solutions.

The channel can be described by a square lattice **undirected graphical model**.

2D channel capacity – graphical model



The variables are binary $x_{\ell,j} \in \{0,1\}$ and the interactions are pair-wise between adjacent variables. Factors:

$$\psi(x_{\ell,j}, x_{m,n}) = \begin{cases} 0, & x_{\ell,j} = x_{m,n} = 1 \\ 1, & \text{otherwise} \end{cases}$$

2D channel capacity – graphical model

The resulting joint PDF is given by

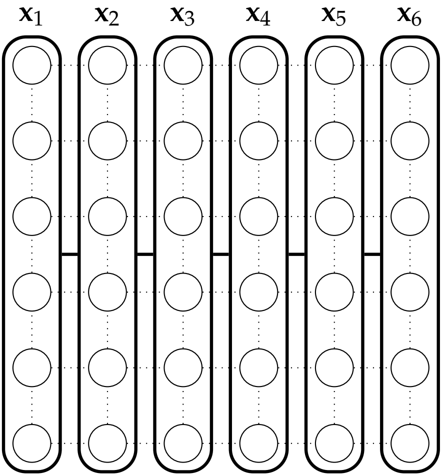
$$p(X_{\mathcal{V}}) = \frac{1}{Z} \prod_{(\ell j, mn) \in \mathcal{E}} \psi(x_{\ell,j}, x_{m,n}),$$

For a channel of dimension $M \times M$ we can write the finite-size **noiseless capacity** as

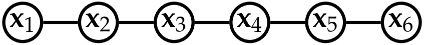
$$C_M = \frac{1}{M^2} \log_2 Z.$$

Unfortunately calculating Z exactly for these types of models is computationally prohibitive, since the complexity is exponential in the number of variables M^2.

2D channel capacity – undirected chain



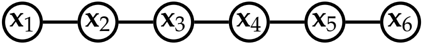
Rewrite the PGM as a high-dimensional **undirected chain** by introducing a new set of variables $\mathbf{x}_k$.



$$\phi(\mathbf{x}_k) = \prod_{j=1}^{M-1} \psi(x_{j+1,k}, x_{j,k}),$$

$$\psi(\mathbf{x}_k, \mathbf{x}_{k-1}) = \prod_{j=1}^{M} \psi(x_{j,k}, x_{j,k-1}).$$

2D channel capacity – SMC algorithm



The **undirected chain** results in the following joint PDF

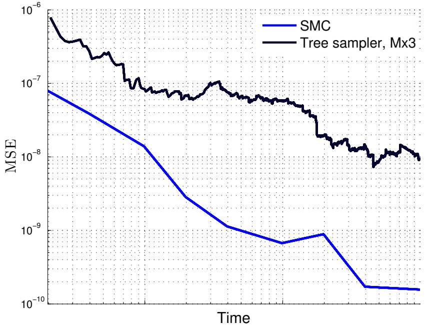
$$p(X_{\mathcal{V}}) = \frac{1}{Z} \prod_{k=1}^{M} \phi(\mathbf{x}_k) \prod_{k=2}^{M} \psi(\mathbf{x}_{k-1}, \mathbf{x}_k).$$

Provides a **natural sequence of target distributions** for SMC!

Sequential decomposition:

$$\begin{aligned} \gamma_1(\mathbf{x}_1) &= \phi(\mathbf{x}_1), \\ \gamma_k(\mathbf{x}_{1:k}) &= \gamma_{k-1}(\mathbf{x}_{1:k-1}) \phi(\mathbf{x}_k) \psi(\mathbf{x}_{k-1}, \mathbf{x}_k). \end{aligned}$$

2D channel capacity – 60×60 example



Our SMC sampler compared to the **tree sampler** by

F. Hamze and N. de Freitas, **From fields to trees**, *In Proceedings of the conference on Uncertainty in Artificial Intelligence (UAI)*, Banff, Canada, July, 2004.

implemented according to

M. Molkaraie and H.-A. Loeliger, **Monte Carlo algorithms for the partition function and information rates of two-dimensional channels**, *IEEE Transactions on Information Theory*, 59(1): 495–503, 2013.

For the 2D channel: **fully adapted** SMC sampler. To sample exactly the $\mathbf{x}_k$'s we use a forward/backward algorithm.

This was just a special case, the important question is, can we do this for a general graphical model?! **Yes!**

Using “standard” SMC for PGMs – the idea

Key idea:

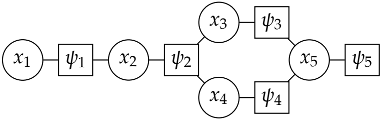
- Perform a **sequential decomposition** of the graphical model.
- Each **subgraph** induces an artificial target distribution.
- Apply SMC to the sequence of artificial target distributions.

Using an artificial sequence of intermediate target distributions for an SMC sampler is a powerful (and **quite possibly underutilised**) idea.

Sequential decomposition of PGMs – pictures

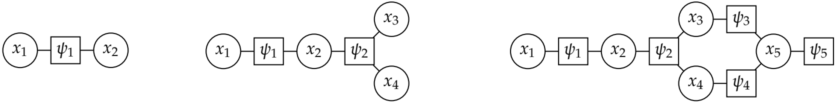
The joint PDF of the set of random variables indexed by $\mathcal{V}$, $X_{\mathcal{V}} \triangleq \{x_1, \ldots, x_{|\mathcal{V}|}\}$

$$p(X_{\mathcal{V}}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(X_C).$$



Example of a sequential decomposition of the above factor graph (the target distributions are built up by adding factors at each iteration),

$\gamma_1(X_{\mathcal{L}_1})$ $\gamma_2(X_{\mathcal{L}_2})$ $\gamma_3(X_{\mathcal{L}_3}) \propto p(X_{\mathcal{V}})$



Sequential decomposition of PGMs – equations

Let $\{\psi_k\}_{k=1}^K$ be a sequence of factors,

$$\psi_k(X_{\mathcal{I}_k}) = \prod_{C \in \mathcal{C}_k} \psi_C(X_C),$$

where $\mathcal{I}_k \subseteq \{1, \ldots, |\mathcal{V}|\}$ is the set of indices in the domain of ψ_k.

The **sequential decomposition** is based on these factors,

$$\gamma_k(X_{\mathcal{L}_k}) \triangleq \prod_{\ell=1}^{k} \psi_\ell(X_{\mathcal{I}_\ell}),$$

where $\mathcal{L}_k \triangleq \bigcup_{\ell=1}^k \mathcal{I}_\ell$.

By construction, $\mathcal{L}_K = \mathcal{V}$ and the joint PDF $p(X_{\mathcal{L}_K}) \propto \gamma_K(X_{\mathcal{L}_K})$.

SMC sampler for graphical models

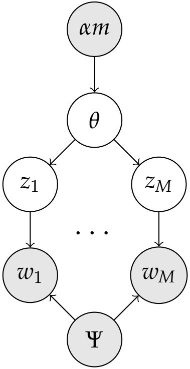
Algorithm SMC sampler for graphical models

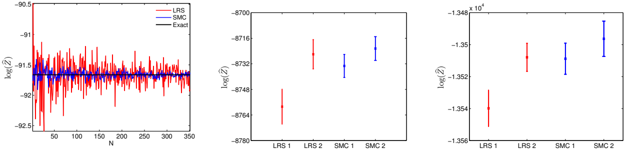
```
1. Initialize ($k = 1$): Draw $X^i_{\mathcal{L}_1} \sim r_1(\cdot)$ and set $w^i_1 = W_1(X^i_{\mathcal{L}_1})$.
2. For $k = 2$ to $K$ do:
    (a) Draw $a^i_k \sim \text{Cat}(\{w^j_{k-1}\}_{j=1}^N)$.
    (b) Draw $\xi^i_k \sim r_k(\cdot | X^{a^i_k}_{\mathcal{L}_{k-1}})$ and set $X^i_{\mathcal{L}_k} = X^{a^i_k}_{\mathcal{L}_{k-1}} \cup \xi^i_k$.
    (c) Set $w^i_k = W_k(X^i_{\mathcal{L}_k})$.
```

Also provides an unbiased estimate of the **partition function**!

Example – Evaluation of topic models

Evaluating Latent Dirichlet Allocation models on heldout documents corresponds to estimating the partition function of a PGM.





(a) Synthetic

(b) PMC

(c) 20 newsgroups

Estimates of the log-likelihood of heldout documents for various datasets.

So what's the catch?

Problems with SMC, it is not enough since:

1. It does not solve the parameter learning problem.
2. The quality of the marginals

$$p(X_{\mathcal{L}_k}) = \int \tilde{\gamma}_K(X_{\mathcal{L}_K}) \mathrm{d}X_{\mathcal{L}_K \setminus \mathcal{L}_k}$$

   deteriorates for $k \ll K$ (particle degeneracy).

(One) solution: Use particle Markov chain Monte Carlo (PMCMC). Allows us to construct high-dimensional MCMC kernels for graphical models.

This allows us to:

1. Simulate, jointly, blocks of variables using an MCMC scheme.
2. Opens up for learning unknown parameters of the model.

PMCMC – Particle Gibbs

```
Algorithm Particle Gibbs, for $r = 1$ to $R$
  1. Given reference trajectory $X_{r-1}$, run conditional SMC
  2. Sample new MCMC sample $X_r$ from particle trajectories
```

Conditional SMC is standard SMC, but we set particle nr N (last) deterministically to the corresponding value in X_{r-1}.

- Leads to a *correct* MCMC method.
- Learning enabled by blocked sampler

$$\theta \sim p(\theta \mid x, y)$$
$$x \sim p(x \mid \theta, y)$$

Christophe Andrieu, Arnaud Doucet and Roman Holenstein, **Particle Markov chain Monte Carlo methods**. *Journal of the Royal Statistical Society: Series B*, 72:269-342, 2010.

Partial blocking (I/II)

Two extremes of how to sample the variables:

1. Simulate all the latent variables $X_{\mathcal{L}_K}$ jointly.
2. Simulate one variable x_j at a time.

With PMCMC we can create algorithms that sits **in between** these two extremes by simulating blocks of variables jointly (**partial blocking**).

Simulate all the latent variables $X_{\mathcal{L}_K}$ jointly.

Partial blocking via PMCMC.

Simulate one variable x_j at a time.

Partial blocking (II/II)

Let $\{\mathcal{V}^m, m \in \{1, \ldots, M\}\}$ be a partition of $\mathcal{V}$.

We could then (ideally) construct a Gibbs sampler simulating from the conditional distributions

$$p(X_{\mathcal{V}^m} | X_{\mathcal{V} \setminus \mathcal{V}^m}) \propto \prod_{C \in \mathcal{C}^m} \psi_C(X_C), \qquad \text{for } m = 1, \ldots, M.$$

where $\mathcal{C}^m = \{C \in \mathcal{C} : C \cap \mathcal{V}^m \neq \emptyset\}$.

Problem: It is (in general) impossible to sample from these conditionals.

(One) solution: Use PMCMC.

Example – Gaussian MRF

Consider a standard square lattice Gaussian MRF of size 10×10,

$$p(X_{\mathcal{V}}, Y_{\mathcal{V}}) \propto \prod_{i \in \mathcal{V}} e^{\frac{1}{2\sigma_i^2}(x_i - y_i)^2} \prod_{(i,j) \in \mathcal{E}} e^{\frac{1}{2\sigma_{ij}^2}(x_i - x_j)^2}$$

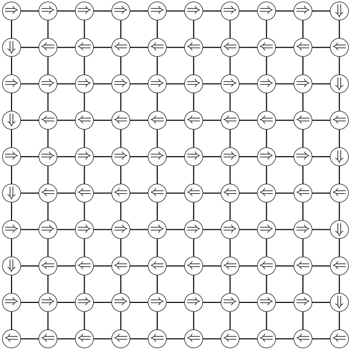
with latent variables $X_{\mathcal{V}} = \{x_1, \dots, x_{100}\}$ and measurements $Y_{\mathcal{V}} = \{y_1, \dots, y_{100}\}$ (simulated with $\sigma_i = 1$ and $\sigma_{ij} = 0.1$).

Goal: Compute the posterior distribution $p(X_{\mathcal{V}} \mid Y_{\mathcal{V}})$.

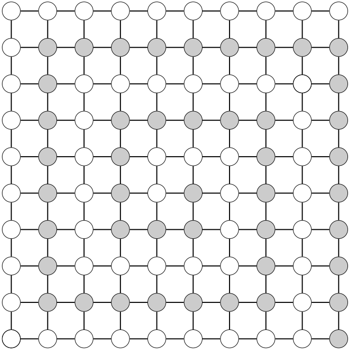
We run four MCMC samplers:

1. Standard one-at-a-time Gibbs
2. Tree sampler (Hamze & de Freitas, 2004)
3. PGAS – fully blocked ($N = 50$)
4. PGAS – partially blocked ($N = 50$)

Example – Gaussian MRF

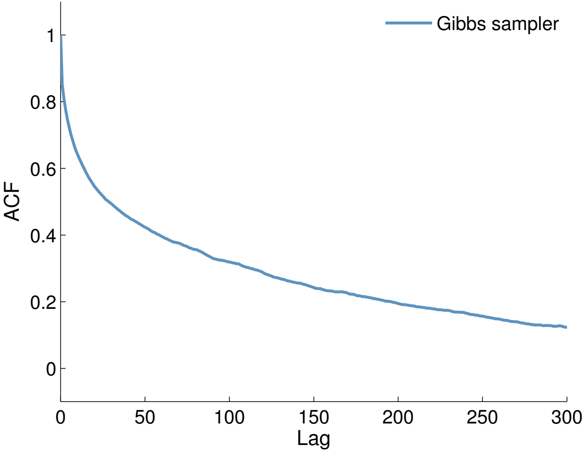


The arrows show the order in which the factors are added.



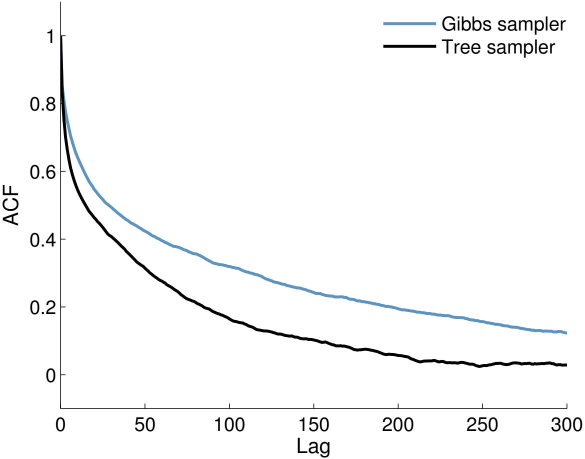
The two block structures used by the tree sampler and PMCMC with partial blocking.

Example – Gaussian MRF



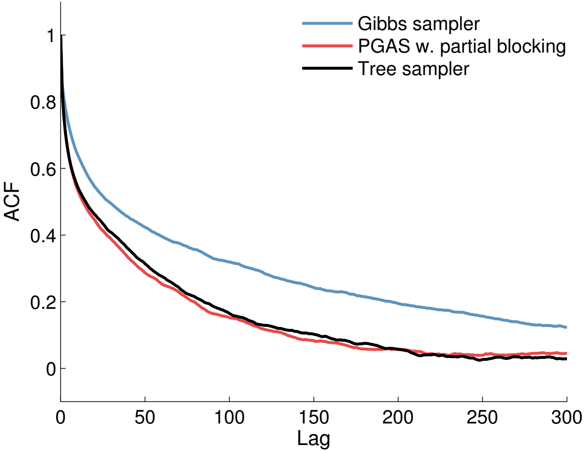
The one-step-at -a-time Gibbs sampler is struggling due to the strong interactions.

Example – Gaussian MRF



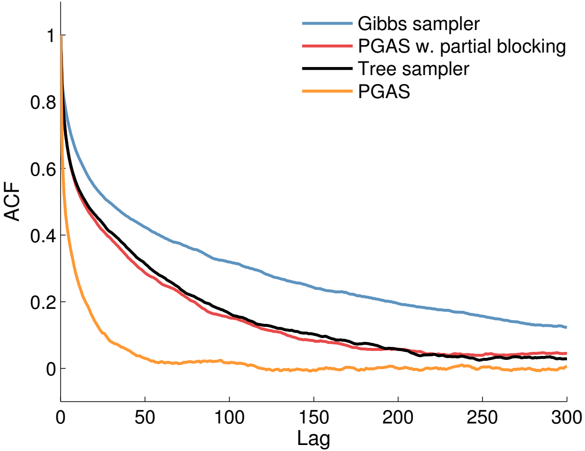
The tree sampler implements an “ideal” partially blocked Gibbs sampler.

Example – Gaussian MRF



PMCMC with partial blocking is an **approximation of the tree sampler**. Already for relatively few particles we obtain a performance similar to the “ideal” tree sampler.

Example – Gaussian MRF



The fully blocked PMCMC performs best, which is not surprising, since it samples all the (dependent) latent variables jointly.

The downside of PMCMC is that it is computationally more expensive.

Conclusions

- We have derived SMC-based inference methods for graphical models of arbitrary topologies with discrete and/or continuous random variables.
- **Key insight:** We exploit decompositions of the graphical model to design efficient SMC and MCMC samplers.
- Examples involving:
  1. estimating the partition function
  2. inferring the latent variables
- If you have interesting and challenging problems involving graphical models, let us know!

Some references

SMC (and PMCMC) methods for graphical models

Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Sequential Monte Carlo for Graphical Models**. *Advances in Neural Information Processing Systems (NIPS) 27*, December, 2014.

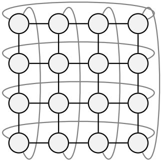
F. Lindsten, A. M. Johansen, C. A. Naesseth, B. Kirkpatrick, T. B. Schön, J. Aston and A. Bouchard-Côté, **Divide-and-Conquer with Sequential Monte Carlo**. *Preprint arXiv:1406.4993*, June, 2014.

Christian A. Naesseth, Fredrik Lindsten and Thomas B. Schön, **Capacity estimation of two-dimensional channels using Sequential Monte Carlo**. *Proceedings of the 2014 IEEE Information Theory Workshop (ITW)*, November, 2014.

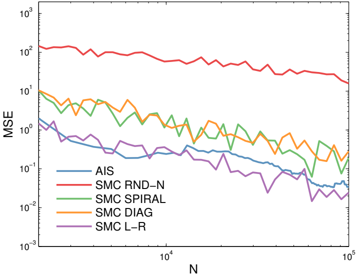
Thank you!!

Example – Classical XY model (I/II)

In statistical mechanics, computing the free energy of a lattice with periodic boundary conditions relates to estimating the partition function of a PGM.

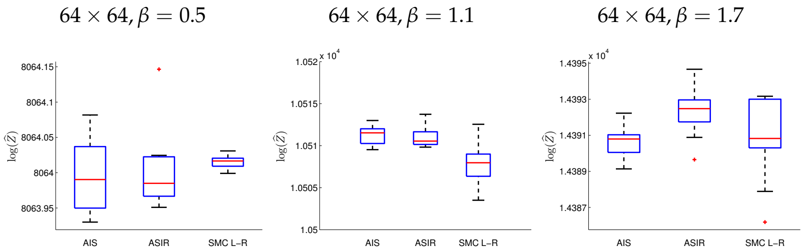


$16 \times 16, \beta = 1.1$



Example – Classical XY model (II/II)

Scaling up to 64×64.



$64 \times 64, \beta = 0.5$

$64 \times 64, \beta = 1.1$

$64 \times 64, \beta = 1.7$

Divide-and-Conquer with SMC – the idea

The sequential decomposition is basically a chain-oriented decomposition of the PGM. This naturally leads to a sequence of distributions suitable for standard SMC samplers.

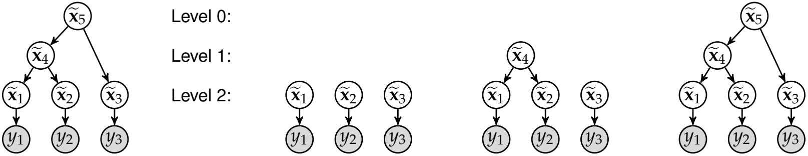
Divide-and-Conquer SMC:

Key idea:

- Consider graph decompositions organised on **trees**.
- Starting from the leaves, define auxiliary target distributions for all nodes of the tree in a **bottom-up** fashion.
- Inference using a **new class** of SMC algorithms.

Tree decomposition of PGMs (I/II)

Hierarchical Bayesian network


$\widetilde{\mathbf{x}}_5$
$\widetilde{\mathbf{x}}_4$
$\widetilde{\mathbf{x}}_1$
$\widetilde{\mathbf{x}}_2$
$\widetilde{\mathbf{x}}_3$
y_1
y_2
y_3
Level 0:
Level 1:
Level 2:

We initialise the D&C-SMC with **independent particle populations** for each leaf in the tree decomposition. These are then merged, resampled and propagated as we move up the tree.

Iter 1: Initialise $(\widetilde{\mathbf{x}}_k^i, \mathbf{w}_k^i)_{i=1}^N$ for $k = 1, 2, 3$.

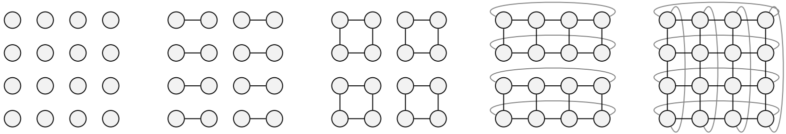
Iter 2: Merge populations 1 and 2 and propagate $\Rightarrow (\widetilde{\mathbf{x}}_{1,2,4}^i, \mathbf{w}_4^i)_{i=1}^N$

Iter 3: Merge populations 3 and 4 and propagate $\Rightarrow (\widetilde{\mathbf{x}}_{1,2,3,4,5}^i, \mathbf{w}_5^i)_{i=1}^N$

Tree decomposition of PGMs (II/II)

Tree decomposition follows naturally when the graphical model is a tree. However, the idea is more generally applicable.

Example: Lattice Markov random field



The subgraphs can be **organised on a tree!**

Divide-and-Conquer SMC

Algorithm D&C-SMC (for node $t \in T$)

```
1. For $c \in \mathcal{C}(t)$:
    1. $(\mathbf{x}_c^i, \mathbf{w}_c^i)_{i=1}^N \leftarrow$ dc_smc($c$).
    2. Resample $(\mathbf{x}_c^i, \mathbf{w}_c^i)_{i=1}^N$ to obtain the equally weighted particle system $(\check{\mathbf{x}}_c^i, 1)_{i=1}^N$.
2. For particle $i = 1, \dots, N$:
    1. Simulate $\widetilde{\mathbf{x}}_t^i \sim q_t(\cdot \mid \check{\mathbf{x}}_{c_1}^i, \dots, \check{\mathbf{x}}_{c_C}^i)$ from some proposal kernel on $\widetilde{\mathsf{X}}_t$, and where $(c_1, c_2, \dots, c_C) = \mathcal{C}(t)$.
    2. Set $\mathbf{x}_t^i = (\check{\mathbf{x}}_{c_1}^i, \dots, \check{\mathbf{x}}_{c_C}^i, \widetilde{\mathbf{x}}_t^i)$.
    3. Compute $\mathbf{w}_t^i = \frac{\gamma_t(\mathbf{x}_t^i)}{\prod_{c \in \mathcal{C}(t)} \gamma_c(\check{\mathbf{x}}_c^i)} \frac{1}{q_t(\widetilde{\mathbf{x}}_t^i \mid \check{\mathbf{x}}_{c_1}^i, \dots, \check{\mathbf{x}}_{c_C}^i)}$.
3. Return $(\mathbf{x}_t^i, \mathbf{w}_t^i)_{i=1}^N$.
```

- Generalises the SMC framework (std SMC recovered if T is a chain).
- Consistent and gives an unbiased estimate of the partition function.

Example – NY maths test (model)

Data Table of test results (278 399 instances), with school code, year, number of students tested in that year and school, and the number students that passed.

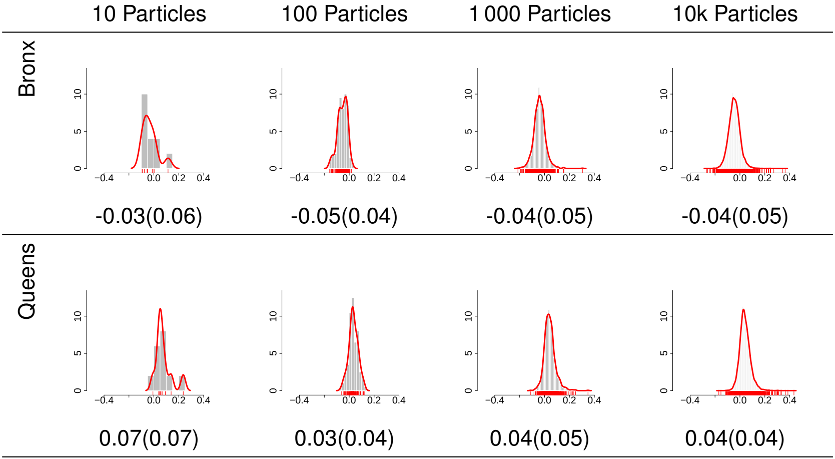
Structure We organise the data into a tree with the following form: NYC (root), borough of the school district, school district, school, year.

Parameters

- Observations at the leaf (binomial $p_t = \mathrm{logistic}(\theta_t)$).
- Internal nodes $\theta_{t'} = \theta_t + \Delta_e$, with $\Delta_e \sim \mathrm{N}(0, \sigma_e^2)$.
- Hyperparameters $\sigma_e^2 \sim \mathrm{Exp}(1)$.

After marginalization of interior nodes, the dimensionality of the *remaining* parameters in the model is 3 555.

Example – NY maths test (results, I/II)



	10 Particles	100 Particles	1 000 Particles	10k Particles
Bronx	-0.03(0.06)	-0.05(0.04)	-0.04(0.05)	-0.04(0.05)
Queens	0.07(0.07)	0.03(0.04)	0.04(0.05)	0.04(0.04)

Posterior distribution of $\delta_e =$"difference in logistic(θ) along edge e" for two boroughs (rows) and four computational regimes (columns), with mean and std dev below each histogram.

Example – NY maths test (results, II/II)

We compare our D&C-SMC (implemented in Java) to Hamiltonian Monte Carlo (Stan, implemented in C++).

Similar posterior approximation accuracy.

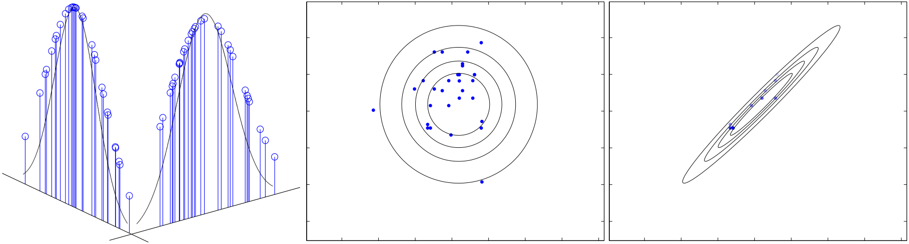
Method	Iterations/Particles	Runtime
D&C-SMC	1000	39 s
HMC (Stan)	2000 (50% burn-in)	3860 s (64 min)

Node	Stan	D&C-SMC	Speedup
NY_Manhattan	0.17	15.96	93.89
NY_Bronx	0.05	8.12	165.69
NY_Kings	0.18	6.52	36.22
NY_Queens	0.07	14.01	209.05
NY_Richmond	0.05	25.50	481.17

The effective samples per second and speedup.

D&C-SMC extensions (I/III)

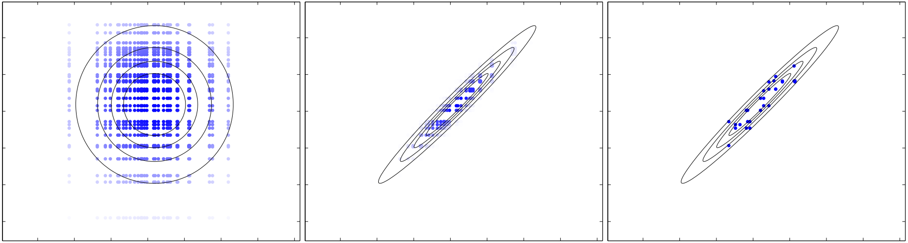
D&C “Sampling Importance Resampling”



$\prod_{c\in\mathcal{C}(t)} \widehat{\gamma}_c^N(\mathbf{x}_c)$ → Resampling → $(\check{\mathbf{x}}^i_{c_1},\dots,\check{\mathbf{x}}^i_{c_C})_{i=1}^N$ → Weighting → $(\mathbf{x}_t^i,\mathbf{w}_t^i)_{i=1}^N$

D&C-SMC extensions (II/III)

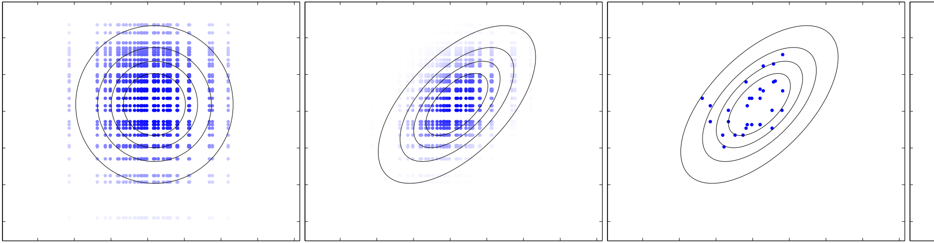
D&C-SMC: Auxiliary mixture sampling



$\prod_{c \in \mathcal{C}(t)} \widehat{\gamma}_c^N(\mathbf{x}_c)$ → (Auxiliary) Weighting → Resampling

D&C-SMC extensions (III/III)

D&C-SMC: Auxiliary mixture sampling + Tempering



$\prod_{c\in\mathcal{C}(t)} \widehat{\gamma}_c^N(\mathbf{x}_c)$ → (Auxiliary) Weighting → Tempering