Towards Automated Sequential Monte Carlo for Probabilistic Graphical Models

Christian A. Naesseth
Div. of Automatic Control
Linköping University
Linköping, Sweden
chran60@isy.liu.se

Fredrik Lindsten
Dept. of Engineering
The University of Cambridge
Cambridge, UK
fsml2@cam.ac.uk

Thomas B. Schön
Dept. of Information Technology
Uppsala University
Uppsala, Sweden
thomas.schon@it.uu.se

Abstract

We revisit the idea of using sequential Monte Carlo (SMC) for inference in general probabilistic graphical models. By constructing a sequence of auxiliary target distributions (also known as a sequential decomposition) based on the graph structure we can run a standard SMC sampler on the graph. In this paper we study the impact of the sequential decomposition on the accuracy of the SMC method by computing the asymptotic variance of the estimator as a function of the decomposition. In general the variance will be intractable, so we propose to use a proxy Gaussian Markov random field with a structure that is identical to that of the original problem. Furthermore, based on these results we propose and evaluate some heuristics for automated SMC inference on any given graph structure.

1 Introduction

Bayesian inference in statistical models involving a large number of latent random variables is in general a difficult problem requiring approximate inference. This renders methods that are capable of efficiently utilizing structure important tools when performing statistical inference. Probabilistic Graphical Models (PGMs) are an intuitive and useful way to represent and make use of underlying structure in probability distributions with many interesting areas of applications [Jordan, 2004].

Our main contribution is a way of efficiently evaluating heuristics for picking the sequence of target distributions in an sequential Monte Carlo (SMC) method without running the actual algorithm. We study the asymptotic variance of the normalization constant estimate as a function of the sequence of distributions. We also propose a class of heuristics for automated SMC inference in PGMs and evaluate these numerically.

In the paper by Ritchie et al. [2015], the authors consider randomizing the order of execution in SMC for probabilistic programs. The method considers all nodes that can be added at any time step, e.g. children with all parents already added in a Bayesian network, and then proceeds to uniformly at random choose the next random variable to add/sample. Thus it bears some resemblance to our *random neighbour* heuristic proposed in [Naesseth et al., 2014], studied further in this paper.

1.1 Graphical Models and SMC

We will consider probabilistic graphical models with pair-wise interaction

$$\bar{\pi}(x_{1:n}) = \frac{1}{Z_\pi}\pi(x_{1:n}) = \frac{1}{Z_\pi}\prod_{i\in\mathcal{V}}\phi(x_i)\prod_{(i,j)\in\mathcal{E}}\psi(x_i,x_j), \tag{1}$$

where the graph, $G = \{\mathcal{V},\mathcal{E}\}$, is described by its edge set $\mathcal{E}$ and vertex set $\mathcal{V} = \{1,\dots,n\}$ and $Z_\pi = \int \pi(x_{1:n})\mathrm{d}x_{1:n}$ is the normalization constant. The goal is then to perform inference with

respect to $\bar{\pi}(x_{1:n})$. Normally this is intractable and we must resort to approximation methods. We will in this paper specifically study the sequential Monte Carlo approach that we proposed in Naesseth et al. [2014].

Our method, hereafter referred to as SMC4PGM, is a standard SMC method for an auxiliary sequence of (unnormalized) target measures $\pi_k, k = 1, \ldots, n$, and their normalized counterparts $\bar{\pi}_k$, with the constraint that $\pi_n = \pi$. The design of this sequence, or equivalently the graph's *sequential decomposition* [Naesseth et al., 2014], is based on a permutation of the indices $1, \ldots, n$ denoted by $\sigma_n = (\sigma(1), \ldots, \sigma(n))$, i.e. $\sigma(i) \neq \sigma(j), i \neq j$ and $\sigma(i) \in \mathcal{V}$. We also write $\sigma_k = (\sigma(1), \ldots, \sigma(k))$ for the list comprising the first k indices of the permutation. In what follows we will sometimes also make use of the sequence σ_k as a (unordered) set. Now, given this permutation we can define the unnormalized target distributions of the SMC sampler as follows:

$$\pi_k(x_{\sigma_k}) = \prod_{i \in \sigma_k} \phi(x_i) \prod_{(i,j) \in \mathcal{E}_k} \psi(x_i, x_j), \tag{2}$$

where $x_{\sigma_k} = (x_{\sigma(1)}, \ldots, x_{\sigma(k)})^\top$, $\mathcal{E}_k = \{(i,j) : i,j \in \sigma_k \text{ and } (i,j) \in \mathcal{E}\}$, and as required $\pi_n(x_{\sigma_n}) = \pi(x_{1:n})$. We make the assumption that these sequences are all normalizable, i.e. $\int \pi_k(x_{\sigma_k}) \mathrm{d}x_{\sigma_k} < \infty$. There are ways of getting around this restriction, see e.g. Naesseth et al. [2014], but we will for simplicity consider only the normalizable case here. Then given resampling weights $\nu_k(x_{\sigma_{k-1}})$ and proposals $q_k(x_{\sigma(k)} \mid x_{\sigma_{k-1}})$ we run a standard auxiliary SMC method targeting $\pi_1, \ldots, \pi_n$. For a more thorough introduction to SMC4PGM see Naesseth et al. [2014] and for more indepth information on SMC samplers in general we refer to the book edited by Doucet et al. [2001] or the tutorial by Doucet and Johansen [2011].

The problem with this algorithm is that it relies on the selection of the ordering or the permutation of the vertices, i.e. the σ_k's. The accuracy of our estimates is highly dependent on this permutation. We will in the rest of this paper study the impact of the permutation and suggest some (greedy) heuristics that can be of interest when trying to perform approximate inference in PGMs.

2 On Sequential Decomposition and Asymptotic Variance

A difficulty in studying the impact of the permutation of the vertices used in the sequential decomposition is that the asymptotic variance of the SMC sampler will depend (intractably) on several parameters; the specific model under study, the implementation choices of the sampler, and the estimand of interest. In order to obtain practical and interpretable results, we will make use of a sequence of simplifications, as detailed below.

2.1 Fully Adapted SMC and Normalizing Constant Estimates

The choice of proposal distributions affect the efficiency of SMC. We assume that a so-called *fully adapted auxiliary* SMC sampler [Pitt and Shephard, 1999] is used. This implementation is known to be locally optimal (it minimizes the variance of the incremental importance weights), so practical SMC implementations should ideally be close to the fully adapted sampler, making it a suitable choice for this study. The fully adapted sampler uses a proposal density and adjustment weights,

$$q_k(x_{\sigma(k)} \mid x_{\sigma_{k-1}}) = \frac{\pi_k(x_{\sigma_k})}{\pi_{k-1}(x_{\sigma_{k-1}})}, \qquad \nu_k(x_{\sigma_{k-1}}) = \int q_k(x_{\sigma(k)} \mid x_{\sigma_{k-1}}) \mathrm{d}x_{\sigma(k)}, \tag{3}$$

respectively.

Furthermore, the asymptotic variance of an SMC estimator of some estimand $\mathbb{E}[\varphi(x_{1:n})]$ will depend on the test function φ. To obtain a function-free criterion for analysing the impact of the permutation we will instead study the SMC estimator of the normalising constant Z_π,

$$Z_\pi^N := \prod_{k=1}^{n-1} \left\{ \frac{1}{N} \sum_{i=1}^{N} \nu_{k+1}(x_{\sigma_k}^i) \right\},$$

where $\{x_{\sigma_k}^i\}_{i=1}^N$ are the particles generated at iteration k. The normalizing constant estimate is of significant interest on its own and, indeed, it is also commonly used as an indicator of the overall efficiency of the SMC implementation.

The estimator Z_π^N is known to be unbiased. Furthermore, due to the construction by Johansen and Doucet [2008], we can use standard theoretical results from the SMC literature [Chopin, 2004, Del Moral, 2004] to give expressions for the asymptotic variance of Z_π^N:

$$\mathrm{Var}\left(\frac{Z_\pi^N}{Z_\pi}\right) = \sum_{k=1}^{n-1} \int \frac{\bar{\pi}(x_{\sigma_k})^2}{\bar{\pi}_k(x_{\sigma_k})} \mathrm{d}x_{\sigma_k} - 1. \tag{4}$$

2.2 Gaussian MRF

The variance (4) will in general be intractable to compute. In order to derive exact, computable, expressions for the asymptotic variance of SMC inference on a graphical model with an arbitrary permutation, we therefore replace our actual model of interest with a proxy Gaussian MRF with identical structure. Assume that the graph $G = \{\mathcal{V}, \mathcal{E}\}$ is known. Then we propose to use the following Gaussian MRF as a proxy to evaluate a sequential decomposition:

$$\bar{\pi}(x_{1:n}) \propto \prod_{i \in \mathcal{V}} e^{-\frac{1}{2}\tau_i x_i^2} \prod_{(i,j)\in\mathcal{E}} e^{-\frac{1}{2}\lambda_{ij}(x_i - x_j)^2}, \tag{5}$$

where $\tau_i, \lambda_{ij} = \lambda_{ji}$ are fixed (known) parameters. Typically, for simplicity, we will let $\tau_i = \tau, \forall i$, and $\lambda_{ij} = \lambda, \forall (i,j)$. Based on the proxy model we can state the following result.

Proposition 1. *Assume a Gaussian MRF, defined by* (5)*, and a permutation σ_n. The asymptotic variance of the the fully adapted auxiliary SMC sampler* (3) *applied to this problem is given by*

$$\mathrm{Var}\left(\frac{Z_\pi^N}{Z_\pi}\right) = \sum_{k=1}^{n-1} \frac{|\Lambda_{n,\sigma_k}|}{\sqrt{|\Lambda_{\sigma_k}||2\Lambda_{n,\sigma_k} - \Lambda_{\sigma_k}|}} - 1, \tag{6}$$

with $\bar{\pi}(x_{\sigma_k}) = \mathcal{N}(0, \Lambda_{n,\sigma_k}^{-1})$, $\bar{\pi}_k(x_{\sigma_k}) = \mathcal{N}(0, \Lambda_{\sigma_k}^{-1})$, and $|\cdot|$ denoting the determinant.

Proof. The result follows by noting that (5) is multivariate Gaussian, i.e. $\mathcal{N}(0, \Lambda_{n,\sigma_n}^{-1})$, and using standard properties of multivariate normals when calculating (4). □

Remark 1. Λ_{n,σ_n} and Λ_{σ_k} are given by the graph, the model parameters, and the permutation. Then, Λ_{n,σ_k} can be computed by first inverting Λ_{n,σ_n}, then extracting and inverting the correct covariance matrix of the marginal distribution.

For large graphs, this result in itself does not allow us to pick an optimal ordering, since optimizing (6) with respect to σ_n is a combinatorial problem. However, the result enables us to compare two different permutations without actually running the computationally demanding algorithm. Furthermore, we can see from (4) that we should try to keep $\bar{\pi}_k(x_{\sigma_k})$ as close to the exact marginal, $\bar{\pi}(x_{\sigma_k})$, as possible. Next we will detail some heuristics to automate the choice of a good permutation for general probabilistic graphical models.

2.3 Heuristics

Picking the *best* sequential decomposition, or permutation, would in general require us to evaluate all $n!$ possible cases of (6). This is obviously computationally prohibitive for any reasonable size n. Due to this we will consider and compare some heuristics for deciding on the permutation.

We define a class of heuristics, to decide the permutation by considering a sequence of weights $W_k = (w_k(1), \ldots, w_k(n)),\ k = 1, \ldots, n-1$ and greedily picking the index of the maximizing entry of W_k at each iteration k. This means that our permutation is constructed as follows:

$$\sigma(k) = \underset{i \in \{1:n\}\setminus\sigma_{k-1}}{\mathrm{argmax}} \; w_k(i), \tag{7}$$

where the sequence W_k is defined by the user and could depend on, amongst other things, previous choices and structure of the graph. We will here consider a class of heuristics defined as follows:

$\mathbf{H}(\mathbf{a}, \mathbf{b}, \mathbf{c})$: For all $i \in \sigma_{k-1}$ we set $w_k(i) = -\infty$ and for the rest we set

$$w_k(i) = a \cdot w_{k-1}(i) + b \cdot \mathrm{card}\{j \in \sigma_{k-1} : (i,j) \in \mathcal{E}\} - c \cdot \mathrm{card}\{j \in \mathcal{V}\setminus(\sigma_{k-1} \cup \{i\}) : (i,j) \in \mathcal{E}\}$$

Furthermore, we also consider a simple "random neighbour" heuristic Naesseth et al. [2014]:

RND-Ne: Simulate $i^* \sim \mathcal{U}\{i \in \Gamma(\sigma_{k-1}) \backslash \sigma_{k-1}\}$ and set $w_k(i^*) = 1,\ w_k(i) = 0,\ i \neq i^*$.

Note that a, b, and c are positive tuning variables, card denotes set cardinality, and $\Gamma(\cdot)$ gives all unique neighbors of its argument in the graph G. Furthermore, for all heuristics we initially pick the node with the lowest degree as $\sigma(1)$, i.e. we let $w_1(i) = \text{card}\{\Gamma(i)\}^{-1}$.

The more informed heuristics $\text{H}(a, b, c)$ depend on the intuition that the similarity between $\bar{\pi}(x_{\sigma_k})$ and $\bar{\pi}_k(x_{\sigma_k})$ is strongly related to how many connections there are between σ_{k-1} and $\sigma(k)$ (controlled by b) and how few there are between $\sigma(k)$ and $\mathcal{V} \setminus \sigma_k$ (controlled by c). The a parameter, which aggregates previous weights, comes from SMC intuition that long-ranging connections are undesirable because of path degeneracy issues.

Size	L-R	$\text{H}(0, 10, 1)$
10×10	29.4	32.7
15×15	95.6	104.5

Table 1: Asymptotic variance results for a square grid with $\tau = \lambda = 1$ with the L-R heuristic.

3 Numerical Results

We begin by evaluating the heuristics, as well as a completely random order (RND), on random graphs of various moderate sizes. Given a size n, and thus $\mathcal{V} = \{1, \ldots, n\}$, we add an edge between vertices i and j with a probability p. In Tables 2 and 3 we report results for sparse ($p = 0.08$) and dense ($p = 0.6$) graphs, respectively. We have also tried other settings of λ, τ, p and n with similar qualitative results (not reported here). Note that lower is better, thus the clear winner overall is the combination $b = 10$ and $c = 1$. For each setting we generated 100 independent random graphs and calculated minimum, median and maximum variance for each heuristic.

Type	a	b	c	Min	Median	Max
$\text{H}(a, b, c)$	0	1	0	14.6	30.1	51.0
$\text{H}(a, b, c)$	0	1	1	27.0	52.2	174.0
$\text{H}(a, b, c)$	0	10	1	**13.4**	**24.6**	**43.3**
$\text{H}(a, b, c)$	0	0.1	1	51.2	100.9	265.7
$\text{H}(a, b, c)$	1	1	0	23.7	42.0	69.6
$\text{H}(a, b, c)$	1	10	0	24.5	40.2	69.6
$\text{H}(a, b, c)$	1	0.1	0	29.9	45.3	71.6
RND-Ne	-	-	-	46.0	85.4	182.1
RND	-	-	-	171.7	440.4	1622.0

Table 2: Asymptotic variance (min/median/max) for a random graph with $n = 50$, $p = 0.08$ and $\tau = \lambda = 1$.

Type	a	b	c	Min	Median	Max
$\text{H}(a, b, c)$	0	1	0	419.6	517.8	631.4
$\text{H}(a, b, c)$	0	1	1	542.7	737.9	1479.7
$\text{H}(a, b, c)$	0	10	1	**393.0**	**487.3**	**600.9**
$\text{H}(a, b, c)$	0	0.1	1	745.3	1173.9	2614.4
$\text{H}(a, b, c)$	1	1	0	430.7	542.7	646.9
$\text{H}(a, b, c)$	1	10	0	439.6	535.0	646.9
$\text{H}(a, b, c)$	1	0.1	0	444.6	548.3	710.6
RND-Ne	-	-	-	1206.5	1877.7	4570.2
RND	-	-	-	1116.2	1910.9	4353.9

Table 3: Asymptotic variance (min/median/max) for a random graph with $n = 50$, $p = 0.6$ and $\tau = \lambda = 1$.

Furthermore, for small enough n we can enumerate all possible orderings and find the optimal value. Results in this case are given in Figure 1 for $n = 6$ and the heuristic with $a = 0$, $b = 10$, $c = 1$. We can see that the best heuristic from the previous experiment actually does find the optimal ordering on many of the generated random graphs. However, we do not expect this to happen in general for graphs of realistic sizes.

A graph often used in applications is the lattice graph with nearest neighbour interaction, examples include images and Ising models. Here we compare the efficient left-right (L-R) heuristic of Naesseth et al. [2014], specifically designed for this special case, to the general purpose heuristic above. Results for this can be seen in Table 1.

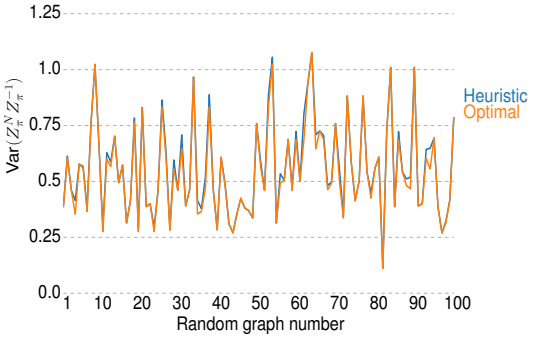

Figure 1: Asymptotic variance results for a random graph with $n = 6$, $p = 0.5$ and $\tau = \lambda = 1$.

Acknowledgments

This work was supported by: Learning of complex dynamical systems (Contract number: 637-2014-466) and Probabilistic modeling of dynamical systems (Contract number: 621-2013-5524), both funded by the Swedish Research Council.

References

M. I. Jordan. Graphical models. *Statistical Science*, 19(1):140–155, 2004.

Daniel Ritchie, Ben Mildenhall, Noah D. Goodman, and Pat Hanrahan. Controlling procedural modeling programs with stochastically-ordered sequential monte carlo. *ACM Trans. Graph.*, 34 (4):105:1–105:11, July 2015. ISSN 0730-0301.

Christian A. Naesseth, Fredrik Lindsten, and Thomas B Schön. Sequential Monte Carlo for Graphical Models. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems (NIPS) 27*, pages 1862–1870. Curran Associates, Inc., 2014.

A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer, New York, 2001.

A. Doucet and A. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In D. Crisan and B. Rozovskii, editors, *The Oxford Handbook of Nonlinear Filtering*. Oxford University Press, 2011.

M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

Adam M. Johansen and Arnaud Doucet. A note on auxiliary particle filters. *Statistics & Probability Letters*, 78(12):1498–1504, 2008.

N. Chopin. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *The Annals of Statistics*, 32(6):2385–2411, 2004.

P. Del Moral. *Feynman-Kac Formulae - Genealogical and Interacting Particle Systems with Applications*. Probability and its Applications. Springer, 2004.