Elements of Sequential Monte Carlo
Other titles in Foundations and Trends® in Machine Learning

*Computational Optimal Transport*
Gabriel Peyre and Marco Cuturi

*An Introduction to Deep Reinforcement Learning*
Vincent Francois-Lavet, Peter Henderson, Riashat Islam, Marc G. Bellemare and Joelle Pineau
ISBN: 978-1-68083-538-0

*An Introduction to Wishart Matrix Moments*
Adrian N. Bishop, Pierre Del Moral and Angele Niclas

*A Tutorial on Thompson Sampling*
Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband and Zheng Wen
Elements of Sequential Monte Carlo

Christian A. Naesseth
Columbia University
christian.a.naesseth@columbia.edu

Fredrik Lindsten
Linköping University
fredrik.lindsten@liu.se

Thomas B. Schön
Uppsala University
thomas.schon@it.uu.se

Full text available at: http://dx.doi.org/10.1561/2200000074
Foundations and Trends® in Machine Learning
Volume 12, Issue 3, 2019
Editorial Board

Editor-in-Chief
Michael Jordan
University of California, Berkeley
United States

Editors
Peter Bartlett
UC Berkeley
Yoshua Bengio
Université de Montréal
Avrim Blum
Toyota Technological Institute
Craig Boutilier
University of Toronto
Stephen Boyd
Stanford University
Carla Brodley
Northeastern University
Inderjit Dhillon
Texas at Austin
Jerome Friedman
Stanford University
Kenji Fukumizu
ISM
Zoubin Ghahramani
Cambridge University
David Heckerman
Amazon
Tom Heskes
Radboud University
Geoffrey Hinton
University of Toronto
Aapo Hyvarinen
Helsinki IIT
Leslie Pack Kaelbling
MIT
Michael Kearns
UPenn
Daphne Koller
Stanford University
John Lafferty
Yale
Michael Littman
Brown University
Gabor Lugosi
Pompeu Fabra
David Madigan
Columbia University
Pascal Massart
Université de Paris-Sud
Andrew McCallum
University of Massachusetts Amherst
Marina Meila
University of Washington
Andrew Moore
CMU
John Platt
Microsoft Research
Luc de Raedt
KU Leuven
Christian Robert
Paris-Dauphine
Sunita Sarawagi
IIT Bombay
Robert Schapire
Microsoft Research
Bernhard Schölkopf
Max Planck Institute
Richard Sutton
University of Alberta
Larry Wasserman
CMU
Bin Yu
UC Berkeley

Full text available at: http://dx.doi.org/10.1561/2200000074
Editorial Scope

Topics

Foundations and Trends® in Machine Learning publishes survey and tutorial articles in the following topics:

- Adaptive control and signal processing
- Applications and case studies
- Behavioral, cognitive and neural learning
- Bayesian learning
- Classification and prediction
- Clustering
- Data mining
- Dimensionality reduction
- Evaluation
- Game theoretic learning
- Graphical models
- Independent component analysis
- Inductive logic programming
- Kernel methods
- Markov chain Monte Carlo
- Model choice
- Nonparametric methods
- Online learning
- Optimization
- Reinforcement learning
- Relational learning
- Robustness
- Spectral methods
- Statistical learning theory
- Variational inference
- Visualization

Information for Librarians

Foundations and Trends® in Machine Learning, 2019, Volume 12, 6 issues. ISSN paper version 1935-8237. ISSN online version 1935-8245. Also available as a combined paper and online subscription.
Contents

1 Introduction 2
  1.1 Historical Background ................................. 3
  1.2 Probabilistic Models and Target Distributions ........ 3
  1.3 Applications ............................................ 9
  1.4 Example Code ........................................... 12
  1.5 Outline .................................................. 13

2 Importance Sampling to Sequential Monte Carlo 14
  2.1 Importance Sampling ..................................... 14
  2.2 Sequential Monte Carlo .................................. 21
  2.3 Analysis and Convergence ............................... 31

3 Learning Proposals and Twisting Targets 37
  3.1 Designing the Proposal Distribution .................... 37
  3.2 Adapting the Target Distribution ........................ 51
  3.A Taylor and Unscented Transforms ....................... 61

4 Nested Monte Carlo: Algorithms and Applications 64
  4.1 The Unbiasedness of the Normalization Constant Estimate 65
  4.2 Particle Metropolis-Hastings ............................ 67
  4.3 Proper Weights and Sequential Monte Carlo ............ 75
  4.4 Distributed Sequential Monte Carlo ..................... 81
  4.A Proof of Unbiasedness ................................. 87
5 Conditional SMC: Algorithms and Applications 90
  5.1 Conditional Sequential Monte Carlo 91
  5.2 The Unbiasedness of the Inverse Normalization Constant Estimate 95
  5.3 The Expected SMC Empirical Distribution 95
  5.4 Parallelization 96
  5.5 Evaluating Inference 98
  5.A Proof of Unbiasedness 103
  5.B Proof of SMC Distribution Representation 104
  5.C Jeffreys Divergence Bound 106

6 Discussion 107

References 111
Elements of Sequential Monte Carlo

Christian A. Naesseth\(^1\), Fredrik Lindsten\(^2\) and Thomas B. Schön\(^3\)

\(^1\)Columbia University; christian.a.naesseth@columbia.edu
\(^2\)Linköping University; fredrik.lindsten@liu.se
\(^3\)Uppsala University; thomas.schon@it.uu.se

ABSTRACT
A core problem in statistics and probabilistic machine learning is to compute probability distributions and expectations. This is the fundamental problem of Bayesian statistics and machine learning, which frames all inference as expectations with respect to the posterior distribution. The key challenge is to approximate these intractable expectations. In this tutorial, we review sequential Monte Carlo (SMC), a random-sampling-based class of methods for approximate inference. First, we explain the basics of SMC, discuss practical issues, and review theoretical results. We then examine two of the main user design choices: the proposal distributions and the so called intermediate target distributions. We review recent results on how variational inference and amortization can be used to learn efficient proposals and target distributions. Next, we discuss the SMC estimate of the normalizing constant, how this can be used for pseudo-marginal inference and inference evaluation. Throughout the tutorial we illustrate the use of SMC on various models commonly used in machine learning, such as stochastic recurrent neural networks, probabilistic graphical models, and probabilistic programs.
A key strategy in machine learning is to break down a problem into smaller and more manageable parts, then process data or unknown variables recursively. Well known examples of this are message passing algorithms for graphical models and annealing for optimization or sampling. Sequential Monte Carlo (SMC) is a class of methods that are tailored to solved statistical inference problems recursively. These methods have mostly received attention in the signal processing and statistics communities. With well over two decades of research in SMC, they have enabled inference in increasingly complex and challenging models. Recently, there has been an emergent interest in this class of algorithms from the machine learning community. We have seen applications to probabilistic programming (Wood et al., 2014), variational inference (VI) (Maddison et al., 2017; Naesseth et al., 2018; Le et al., 2018), inference evaluation (Grosse et al., 2015; Cusumano-Towner and Mansinghka, 2017), probabilistic graphical models (PGMs) (Ihler and McAllester, 2009; Naesseth et al., 2014; Paige and Wood, 2016), Bayesian nonparametrics (Fearnhead, 2004) and many other areas.

We provide a unifying view of the SMC methods that have been developed since their conception in the early 1990s (Gordon et al., 1993; Stewart
1.1 Historical Background

SMC methods are generic tools for performing approximate (statistical) inference, predominantly Bayesian inference. They use a weighted sample set to iteratively approximate the posterior distribution of a probabilistic model. Ever since the dawn of Monte Carlo methods (see e.g. Metropolis and Ulam (1949) for an early discussion), random sample-based approximations have been recognized as powerful tools for inference in complex probabilistic models. Parallel to the development of Markov chain Monte Carlo (MCMC) methods (Metropolis et al., 1953; Hastings, 1970), sequential importance sampling (SIS) (Handschin and Mayne, 1969) and sampling/importance resampling (Rubin, 1987) laid the foundations for what would one day become SMC.

SMC methods where initially known as particle filters (Gordon et al., 1993; Stewart and McCarty, 1992; Kitagawa, 1993). Particle filters where conceived as algorithms for online inference in nonlinear state space models (SSMs) (Cappé et al., 2005). Since then there has been a flurry of work applying SMC and particle filters to perform approximate inference in ever more complex models. While research in SMC initially focused on SSMs, we will see that SMC can be a powerful tool in a much broader setting.

1.2 Probabilistic Models and Target Distributions

As mentioned above, SMC methods were originally developed as an approximate solution to the so called filtering problem, which amounts to online inference in dynamical models. Several overview and tutorial articles focus on particle filters, i.e. the SMC algorithms specifically tailored to solve the online filtering problem (Arulampalam et al., 2002; Doucet and Johansen, 2009; Fearnhead and Künsch, 2018). In this tutorial we will take a different view and explain how SMC can be used to solve more general “offline” problems. We shall see how this viewpoint opens up for many interesting applications of SMC in machine learning that do not fall in the traditional
filtering setup, and furthermore how it gives rise to new and interesting design choices. For complementary review and tutorial articles that treat a similar topic see also Del Moral et al. (2006a) and Doucet and Lee (2018).

We consider a generic probabilistic model given by a joint probability distribution function (PDF) of latent variables $x$ and observed data $y$,

$$p(x, y). \quad (1.1)$$

We focus on Bayesian inference, where the key object is the posterior distribution

$$p(x \mid y) = \frac{p(x, y)}{p(y)}, \quad (1.2)$$

where $p(y)$ is known as the marginal likelihood.

The target distributions are a sequence of probability distributions that we recursively approximate using SMC. We define each target distribution $\gamma_t(x_{1:t})$ in the sequence as a joint PDF of latent variables $x_{1:t} = (x_1, \ldots, x_t)$, where $t = 1, \ldots, T$. The PDF is denoted by

$$\gamma_t(x_{1:t}) := \frac{1}{Z_t} \tilde{\gamma}_t(x_{1:t}), \quad t = 1, \ldots, T, \quad (1.3)$$

where $\tilde{\gamma}_t$ is a positive integrable function and $Z_t$ is the normalization constant, ensuring that $\gamma_t$ is indeed a PDF.

We connect the target distributions to the probabilistic model through a requirement on the final target distribution $\gamma_T(x_{1:T})$. We enforce the condition that $\gamma_T(x_{1:T})$ is either equivalent to the posterior distribution, or that it contains the posterior distribution as a marginal distribution. The intermediate target distributions, i.e. $\gamma_t(x_{1:t})$ for $t < T$, are useful only insofar they help us approximate the final target $\gamma_T(x_{1:T})$. This approach is distinct from most previous tutorials on particle filters and SMC that traditionally focus on the intermediate targets, i.e. the filtering distributions. We stress that it is not necessarily the case that $x_{1:T}$ (the latent variables of the target distribution) is equal to $x$ (the latent variables of the probabilistic model of interest), as the former can include additional auxiliary variables.

Below we introduce a few examples of probabilistic models and some straightforward choices of target distributions. We introduce and illustrate our running example which will be used throughout. We will return to the issue of choosing the sequence of intermediate targets in Section 3.2.
State Space Models  The state space model (or hidden Markov model) is a type of probabilistic models where the latent variables and data satisfy a Markov property. For this model we typically have $x = x_{1:T}$. Often the measured data can also be split into a sequence of the same length ($T$) as the latent variables, i.e. $y = y_{1:T}$. The model is defined by a transition PDF $f$ and an observation PDF $g$,

$$x_t | x_{t-1} \sim f(\cdot | x_{t-1}), \quad \text{(1.4a)}$$

$$y_t | x_t \sim g(\cdot | x_t). \quad \text{(1.4b)}$$

The joint PDF is

$$p(x, y) = p(x_1)g(y_1 | x_1) \prod_{t=2}^{T} f(x_t | x_{t-1})g(y_t | x_t), \quad \text{(1.5)}$$

where $p(x_1)$ is the prior on the initial state $x_1$. This class of models is especially common for data that has an inherent temporal structure such as in the field of signal processing. A common choice is to let the target distributions follow the same sequential structure as in Eq. (1.5):

$$\tilde{\gamma}_t(x_{1:t}) = p(x_1)g(y_1 | x_1) \prod_{k=2}^{t} f(x_k | x_{k-1})g(y_k | x_k), \quad \text{(1.6)}$$

which means that the final normalized target distribution satisfies $\gamma_T(x_{1:T}) = p(x | y)$ as required. This is the model class and target distributions which are studied in the classical filtering setup.

Non-Markovian Latent Variable Models  The non-Markovian latent variable models (LVMs) are characterized by either no, or higher order, Markov structure between the latent variables $x$ and/or data $y$. This can be seen as a non-trivial extension of the SSM, see Eq. (1.4), which has a Markov structure. Also for this class of models it is common to have $x = x_{1:T}$ and $y = y_{1:T}$.

Unlike the SSM, the non-Markovian LVM in its most general setting requires access to all previous latent variables $x_{1:t-1}$ to generate $x_t, y_t$

$$x_t | x_{1:t-1} \sim f_t(\cdot | x_{1:t-1}), \quad \text{(1.7a)}$$

$$y_t | x_{1:t} \sim g_t(\cdot | x_{1:t}), \quad \text{(1.7b)}$$
where we again refer to \( f_t \) and \( g_t \) as the transition PDF and observation PDF, respectively. The joint PDF is given by

\[
p(x, y) = p(x_1)g(y_1 | x_1) \prod_{t=2}^{T} f_t(x_t | x_{1:t-1})g_t(y_t | x_{1:t}),
\]

(1.8)

where \( p(x_1) \) is again the prior on \( x_1 \). A typical target distribution is given by

\[
\bar{\gamma}_t(x_{1:t}) = \bar{\gamma}_{t-1}(x_{1:t-1})f_t(x_t | x_{1:t-1})g_t(y_t | x_{1:t}), \quad t > 1,
\]

(1.9)

with \( \bar{\gamma}_1(x_1) = p(x_1)g_1(y_1 | x_1) \). Another option is

\[
\bar{\gamma}_1(x_1) = p(x_1),
\bar{\gamma}_t(x_{1:t}) = \bar{\gamma}_{t-1}(x_{1:t-1})f_t(x_t | x_{1:t-1}), \quad 1 < t < T,
\bar{\gamma}_T(x_{1:T}) = \bar{\gamma}_{T-1}(x_{1:T-1})f_T(x_T | x_{1:T-1}) \prod_{t=1}^{T} g_t(y_t | x_{1:t}).
\]

For both these sequences of target distributions the final iteration \( T \) is the posterior distribution, i.e. \( \gamma_T(x_{1:T}) = p(x_{1:T} | y_{1:T}) = p(x | y) \). However, the former one will often lead to more accurate inferences. This is because we introduce information from the data at an earlier stage in the SMC algorithm.

Throughout the monograph we will exemplify the different methods using a Gaussian special case of Eq. (1.7), see Example 1.1. We let the prior on \( x_1:t \), defined by the transition PDFs \( f_1, \ldots, f_T \), be Markovian and introduce the non-Markov property instead through the observation PDFs \( g_1, \ldots, g_T \).

**Example 1.1** (Non-Markovian Gaussian Sequence Model). As our running example for illustration purposes we use a non-Markovian Gaussian sequence model. It is

\[
x_t | x_{1:t-1} \sim f_t(\cdot | x_{t-1}), \quad y_t | x_{1:t} \sim g_t(\cdot | x_{1:t}),
\]

(1.10)

with observed variables \( y_t \) (data), and where

\[
f_t(x_t | x_{t-1}) = \mathcal{N}(x_t | \phi x_{t-1}, q),
\]

\[
g_t(y_t | x_{1:t}) = \mathcal{N}\left(y_t | \sum_{k=1}^{t} \beta^{t-k} x_k, r\right),
\]
1.2. Probabilistic Models and Target Distributions

where $\mathcal{N}(x \mid \mu, \Sigma)$ denotes a Gaussian distribution on $x$ with mean $\mu$ and (co)variance $\Sigma$. We let the prior at $t = 1$ be $p(x_1) = \mathcal{N}(x_1 \mid 0, q)$. Artificial data was generated using $(\phi, q, \beta, r) = (0.9, 1, 0.5, 1)$. The distribution of interest is the posterior distribution $p(x_{1:T} \mid y_{1:T})$. We illustrate a few sample paths of $y_{1:T}$ in Fig. 1.1 for $T = 100$.

We can adjust the strength of the dependence on previous latent variables in the observations, $y_t$, through the parameter $\beta \in [0, 1]$. If we set $\beta = 0$ we obtain a linear Gaussian SSM, since the data depends only on the most recent latent $x_t$. On the other hand if we let $\beta = 1$, this signifies that $x_k$ for $k < t$ has equally strong effect on $y_t$ as does $x_t$.

Another example of a non-Markovian LVM often encountered in machine learning is the stochastic recurrent neural network (RNN) (Chung et al., 2015; Bayer and Osendorfer, 2014; Fraccaro et al., 2016; Maddison et al., 2017; Naesseth et al., 2018). We define the stochastic RNN below and will return to it again in Chapter 3.

Example 1.2 (Stochastic Recurrent Neural Network). A stochastic RNN is a non-Markovian LVM where the parameters of the transition and observation models are defined by RNNS. A common example is using the conditional Gaussian distribution to define the transition PDF

$$f_t(x_t \mid x_{1:t-1}) = \mathcal{N}(x_t \mid \mu_t(x_{1:t-1}), \Sigma_t(x_{1:t-1})),$$

where the functions $\mu_t(\cdot), \Sigma_t(\cdot)$ are defined by RNNS.
Conditionally independent models  A common model in probabilistic
machine learning is to assume that the datapoints \( y_k \) in the dataset \( y = \{ y_k \}_{k=1}^K \) are conditionally independent given the latent \( x \). This means that the joint PDF is given by

\[
p(x, y) = p(x) \prod_{k=1}^{K} g_k(y_k \mid x),
\]

where \( p(y \mid x) \) is the likelihood. For this class of models it might not be immediately apparent that we can define a useful sequence of target distributions on latent variables \( x_{1:t} \). There is no obvious sequence structure as in the SSM or the non-Markovian LVM. However, as we shall see, we can make use of auxiliary variables to design target distributions that can help with inference.

We will discuss two approaches to design the sequence of target distributions: using data tempering and likelihood tempering, respectively. Both of these will make use of an auxiliary variable technique, where each \( x_t \) is a random variable on the same space as \( x \).

**Data tempering:** Using data tempering we add the data \( y_k \) to the target distribution one by one. In this case the data index \( k \) coincides with the target index \( t \). We define the target distribution

\[
\gamma_t(x_{1:t}) = p(x_t) \prod_{k=1}^{t} g_k(y_k \mid x_t) \cdot \prod_{k=1}^{t-1} s_k(x_k \mid x_{k+1}),
\]

where the distributions \( s_k(x_k \mid x_{k+1}) \) are a design choice, known as **backward kernels** (Chopin, 2002; Del Moral et al., 2006b). With this choice, we have that the marginal distribution of \( x_T \) at the final iteration is exactly the posterior distribution, i.e. \( \gamma_T(x_T) = p(x \mid y) \). In fact, at each step we have that the marginal target distribution is a partial posterior \( \gamma_t(x_t) = p(x \mid y_{1:t}) \).

**Likelihood tempering:** With likelihood tempering, instead of adding the data one by one, we change the likelihood \( p(y \mid x) \) through a sequence of positive variables. We define the target distribution

\[
\tilde{\gamma}_t(x_{1:t}) = p(x_t) p(y \mid x_t)^{\tau_t} \cdot \prod_{k=1}^{t-1} s_k(x_k \mid x_{k+1}),
\]
where \( 0 = \tau_1 < \ldots < \tau_T = 1 \), and again make use of user chosen backward kernels \( s_k(x_k | x_{k+1}) \) (Chopin, 2002; Del Moral et al., 2006b). In this setting all data is considered at each iteration. Since \( \tau_T = 1 \), we have that the final marginal target distribution is again equal to the posterior \( \gamma_T(x_T) = p(x | y) \).

Applying SMC methods to tempered (and similar) target distributions has been studied by e.g. Chopin (2002) and Del Moral et al. (2006b). If the proposal \( q_{k+1}(x_{k+1} | x_k) \) is a Markov kernel with stationary distribution \( \gamma_{k+1}(x_{k+1}) \), then a commonly used backward kernel is \( s_k(x_k | x_{k+1}) = \frac{\gamma_k(x_k)}{\gamma_{k+1}(x_{k+1}) q_{k+1}(x_{k+1} | x_k) \gamma_{k+1}(x_{k+1})} \). We refer to the works of Chopin (2002) and Del Moral et al. (2006b) for a thorough discussion on the choice of backward kernels \( s_k(x_k | x_{k+1}) \). Another well known example is annealed importance sampling by Neal (2001), which uses the backward kernel example above to define the target distributions, but relies on SIS instead of SMC for inference.

**Models and Targets** We have seen several probabilistic models with examples of corresponding target distributions. While not limited to these, this illustrates the wide range of the applicability of SMC. In fact, as long as we can design a sequence of target distributions such that \( \gamma_T \) coincides with the distribution of interest, we can leverage SMC for inference.

### 1.3 Applications

Sequential Monte Carlo and importance sampling methods have already seen a plethora of applications to machine learning and statistical inference problems. Before we turn to the fundamentals of the various algorithms it can be helpful to understand some of these applications. We present and discuss a few select examples of applications of SMC and importance sampling (IS) to probabilistic graphical models, Bayesian nonparametric models, probabilistic programming, and inference evaluation.

**Probabilistic Graphical Models** Probabilistic graphical models (PGM; see e.g. Koller et al. (2009) and Wainwright and Jordan (2008)) are probabilistic models where the conditional independencies in the joint PDF are described by edges in a graph. The graph structure allows for easy and strong control on the type of prior information that the user can express.
The main limitation of the PGM is that exact inference is often intractable and approximate inference is challenging.

The PGM is a probabilistic model where the PDF factorizes according to an underlying graph described by a set of cliques $C \in \mathcal{C}$, i.e. fully connected subsets of the vertices $V \in \mathcal{V}$ where $\mathcal{V}$ contains all individual components of $x$ and $y$. The undirected graphical model can be denoted by

$$ p(x,y) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(x_C), $$

(1.14)

where $x_C$ includes all elements of $x$ and $y$ in the clique $C$, and $Z$ is a normalization constant ensuring that the right hand side is a proper PDF.

SMC methods have recently been successfully applied to the task of inference in general PGMs, see e.g. MacEachern et al. (1999), Chopin (2002), Del Moral et al. (2006b), Ihler and McAllester (2009), Naesseth et al. (2014), Paige and Wood (2016), Lindsten et al. (2017), and Lindsten et al. (2018) for representative examples.

**Probabilistic Programming** Probabilistic programming languages are programming languages designed to describe probabilistic models and to automate the process of doing inference in those models. We can think of probabilistic programming as that of automating statistical inference, particularly Bayesian inference, using tools from computer science and computational statistics. Developing a syntax and semantics to describe and denote probabilistic models and the inference problems we are interested in solving is key to the probabilistic programming language. To define what separates a probabilistic programming language from a standard programming language we quote Gordon et al. (2014): “Probabilistic programs are usual functional or imperative programs with two added constructs: (1) the ability to draw values at random from distributions, and (2) the ability to condition values of variables in a program via observations.” This aligns very well with the notion of Bayesian inference through the posterior distribution Eq. (1.2); through the syntax of the language we define $x$ to be the random values we sample and $y$ our observations that we condition on through the use of Bayes rule. The probabilistic program then essentially defines our joint probabilistic model $p(x, y)$. 

Full text available at: http://dx.doi.org/10.1561/2200000074
1.3. Applications

One of the main challenges of probabilistic programming is to develop algorithms that are general enough to enable inference for any model (probabilistic program) that we could conceivably write down using the language. Recently Wood et al. (2014) have shown that SMC-based approaches can be used as inference back-ends in probabilistic programs.

For a more thorough treatment of probabilistic programming we refer the interested reader to the recent tutorial by Meent et al. (2018) and the survey by Gordon et al. (2014).

**Bayesian nonparametric models** Nonparametric models are characterized by having a complexity which grows with the amount of available data. In a Bayesian context this implies that the usual latent random variables (i.e., parameters) of the model are replaced by latent stochastic processes. Examples include Gaussian processes, Dirichlet processes, and Beta processes; see e.g. Hjort et al. (2010) for a general introduction.

Sampling from these latent stochastic processes, conditionally on observed data, can be done using SMC. To give a concrete example, consider the Dirichlet process mixture model, which is a clustering model that can handle an unknown and conceptually infinite number of mixture components. Let \( y_t, t = 1, 2, \ldots \) be a stream of data. Let \( x_t, t = 1, 2, \ldots \) be a sequence of latent integer-valued random variables, such that \( x_t \) is the index of the mixture component to which datapoint \( y_t \) belongs. A generative representation of the mixture assignment variables is given by

\[
p(x_{t+1} = j \mid x_{1:t}) = \begin{cases} \frac{n_{t,j}}{t+\alpha} & \text{for } j = 1, \ldots, J_t, \\ \frac{\alpha}{t+\alpha} & \text{for } j = J_t + 1, \end{cases}
\]

where \( J_t := \max\{x_{1:t}\} \) is the number of distinct mixture components represented by the first \( t \) datapoints, and \( n_{t,j} := \sum_{k=1}^{t} 1\{x_k = j\} \) is the number of datapoints among \( y_{1:t} \) that belong to the \( j \)th component.

The model is completed by specifying the distribution of the data, conditionally on the mixture assignment variables:

\[
\theta_k \sim F(\theta), \quad k = 1, 2, \ldots
\]

\[
p(y_t \mid x_t, \{\theta_k\}_{k \geq 1}) = G(y_t \mid \theta_{x_t}),
\]

where \( G(\cdot \mid \theta) \) is an emission probability distribution parameterized by \( \theta \) and \( F(\theta) \) is a prior distribution over the mixture parameters \( \theta \).
Introduction

Note that the mixture assignment variables \(x_t, t = 1, 2, \ldots\) evolve according to a latent stochastic process. Solving the clustering problem amounts to computing the posterior distribution of this stochastic process, conditionally on the observed data. One way to address this problem is to use SMC; see Fearnhead (2004) for an efficient implementation tailored to the discrete nature of the problem.

Inference Evaluation An important problem when performing approximate Bayesian inference is to figure out when our approximation is “good enough”? Is it possible to give practical guarantees on the approximation we obtain? We need ways to evaluate how accurate our approximate inference algorithms are when compared to the true target distribution that we are trying to approximate. We will refer to the process of evaluating and validating approximate inference methods as inference evaluation.

Inference evaluation is mainly concerned with measuring how close our approximation is to the true object we are trying to estimate, often a posterior distribution. For simulated data, Grosse et al. (2015) and Grosse et al. (2016) have shown that we can make use of SMC and IS to bound the symmetrized Kullback-Leibler (KŁ) divergence between our approximate posterior and the true posterior. In another related work Cusumano-Towner and Mansinghka (2017) have shown that SMC-based methods show promise in estimating the symmetric KŁ divergence between the approximate posterior and a gold standard algorithm.

1.4 Example Code

We will be making use of inline Python code snippets throughout the manuscript to illustrate the algorithms and methods. Below we summarize the modules that are necessary to import to run the code snippets:

```python
import numpy as np
import numpy.random as npr
from scipy.misc import logsumexp
from scipy.stats import norm
```

Example Code 1.1: Necessary imports for Python code examples.
1.5 Outline

The remainder of this tutorial is organized as follows. In Chapter 2, we first introduce IS, a foundational building block for SMC. Then, we discuss the limitations of IS and how SMC resolves these. Finally, the section concludes with discussing some practical issues and theoretical results relevant to SMC methods.

Chapter 3 is focused on the two key design choices of SMC: the proposal and target distributions. Initially we focus on the proposal, discussing various ways of adapting and learning good proposals that will make the approximation more accurate. Then we discuss the sequence of target distributions; how we can learn intermediate distributions that help us when we try to approximate the posterior.

Chapter 4 focuses on pseudo marginal (PM) methods and other SMC methods that rely on a concept known as proper weights. First, we provide a simple and straightforward proof of the unbiasedness property of the SMC normalization constant estimate. Then, we describe and illustrate the combination of MCMC and SMC methods through PM algorithms. We move on to detail properly weighted SMC, a concept that unites and extends the random weights and nested SMC algorithms. Finally, we conclude the chapter by considering a few approaches for distributed and parallel SMC.

In Chapter 5 we introduce conditional sequential Monte Carlo (CSMC), and related methods for simulating from and computing expectations with respect to a target distribution. First, we introduce the basic CSMC algorithm and provide a straightforward proof of the unbiasedness of the inverse normalization constant estimate. Then, we show how SMC and CSMC can be combined to leverage multi-core and parallel computing architectures in the interacting particle Markov chain Monte Carlo (IPMCMC) algorithm. Finally, we discuss recent applications of CSMC for evaluating approximate inference methods. The tutorial concludes with a discussion and outlook in Chapter 6.


References


References


