vSMC – PARALLEL SEQUENTIAL MONTE CARLO IN C++

Yan Zhou

March 16, 2016

abstract

Sequential Monte Carlo is a family of algorithms for sampling from a sequence of distributions. Some of these algorithms, such as particle filters, are widely used in the physics and signal processing researches. More recent developments have established their application in more general inference problems such as Bayesian modeling.

These algorithms have attracted considerable attentions in recent years as they admit natural and scalable parallelization. However, these algorithms are perceived to be difficult to implement. In addition, parallel programming is often unfamiliar to many researchers though conceptually appealing, especially for sequential Monte Carlo related fields.

A C++ template library is presented for the purpose of implementing general sequential Monte Carlo algorithms on parallel hardware. Two examples are presented: a simple particle filter and a classic Bayesian modeling problem.

1 INTRODUCTION

Sequential Monte Carlo (smc) methods are a class of sampling algorithms that combine importance sampling and resampling. They have been primarily used as "particle filters" to solve optimal filtering problems; see, for example, Cappé, Godsill, and Moulines (2007) and Doucet and Johansen (2011) for recent reviews. They are also used in a static setting where a target distribution is of interest, for example, for the purpose of Bayesian modeling. This was proposed by Del Moral, Doucet, and Jasra (2006b) and developed by Peters (2005) and Del Moral, Doucet, and Jasra (2006a). This framework involves the construction of a sequence of artificial distributions on spaces of increasing dimensions which admit the distributions of interest as particular marginals.

smc algorithms are perceived as being difficult to implement while general tools were not available until the development by Johansen (2009), which provided a general framework for implementing smc algorithms. SMC algorithms admit natural and scalable parallelization. However, there are only parallel implementations of SMC algorithms for many problem specific applications, usually associated with specific SMC related researches. Lee et al. (2010) studied the parallelization of SMC algorithms on GPUs with some generality. There are few general tools to implement SMC algorithms on parallel hardware though multicore CPUs are very common today and computing on specialized hardware such as GPUs are more and more popular.

The purpose of the current work is to provide a general framework for implementing SMC algorithms on both sequential and parallel hardware. There are two main goals of the presented framework. The first is reusability. It will be demonstrated that the same implementation source code can be used to build a serialized sampler, or using different programming models (for example, OpenMP and Intel TBB) to build parallelized samplers for multicore CPUs. They can be scaled for clusters using MPI with few modifications. And with a little effort they can also be used to build parallelized samplers on specialized massive parallel hardware such as GPUs using OpenCL. The second is extensibility. It is possible to write a backend for vSMC to use new parallel programming models while reusing existing implementations. It is also possible to enhance the library to improve performance for specific applications. Almost all components of the library can be reimplemented by users and thus if the default implementation is not suitable for a specific application, they can be replaced while being integrated with other components seamlessly.

2 SEQUENTIAL MONTE CARLO

2.1 SEQUENTIAL IMPORTANCE SAMPLING AND RESAMPLING

Importance sampling is a technique which allows the calculation of the expectation of a function φ with respect to a distribution π using samples from some other distribution η with respect to which π is absolutely continuous, based on the identity,

$$\mathbb{E}_\pi[\varphi(X)] = \int \varphi(x)\pi(x)\,\mathrm{d}\,x = \int \frac{\varphi(x)\pi(x)}{\eta(x)}\eta(x)\,\mathrm{d}\,x = \mathbb{E}_\eta\Big[\frac{\varphi(X)\pi(X)}{\eta(X)}\Big] \tag{1}$$

And thus, let $\{X^{(i)}\}_{i=1}^N$ be samples from η, then $\mathbb{E}_\pi[\varphi(X)]$ can be approximated by

$$\hat{\varphi}_1 = \frac{1}{N}\sum_{i=1}^N \frac{\varphi(X^{(i)})\pi(X^{(i)})}{\eta(X^{(i)})} \tag{2}$$

In practice π and η are often only known up to some normalizing constants, which can be estimated using the same samples. Let $w^{(i)} = \pi(X^{(i)})/\eta(X^{(i)})$, then we have

$$\hat{\varphi}_2 = \frac{\sum_{i=1}^N w^{(i)}\varphi(X^{(i)})}{\sum_{i=1}^N w^{(i)}} \tag{3}$$

or

$$\hat{\varphi}_3 = \sum_{i=1}^{N} W^{(i)}\varphi(X^{(i)}) \tag{4}$$

where $W^{(i)} \propto w^{(i)}$ and are normalized such that $\sum_{i=1}^{N} W^{(i)} = 1$.

Sequential importance sampling (SIS) generalizes the importance sampling technique for a sequence of distributions $\{\pi_t\}_{t\geq 0}$ defined on spaces $\{\prod_{k=0}^{t} E_k\}_{t\geq 0}$. At time $t = 0$, sample $\{X_0^{(i)}\}_{i=1}^{N}$ from η_0 and compute the weights $W_0^{(i)} \propto \pi_0(X_0^{(i)})/\eta_0(X_0^{(i)})$. At time $t \geq 1$, each sample $X_{0:t-1}^{(i)}$, usually termed *particles* in the literature, is extended to $X_{0:t}^{(i)}$ by a proposal distribution $q_t(\cdot|X_{0:t-1}^{(i)})$. And the weights are recalculated by $W_t^{(i)} \propto \pi_t(X_{0:t}^{(i)})/\eta_t(X_{0:t}^{(i)})$ where

$$\eta_t(X_{0:t}^{(i)}) = \eta_{t-1}(X_{0:t-1}^{(i)})q_t(X_{0:t}^{(i)}|X_{0:t-1}^{(i)}) \tag{5}$$

and thus

$$\begin{aligned} W_t^{(i)} \propto \frac{\pi_t(X_{0:t}^{(i)})}{\eta_t(X_{0:t}^{(i)})} &= \frac{\pi_t(X_{0:t}^{(i)})\pi_{t-1}(X_{0:t-1}^{(i)})}{\eta_{t-1}(X_{0:t-1}^{(i)})q_t(X_{0:t}^{(i)}|X_{0:t-1}^{(i)})\pi_{t-1}(X_{0:t-1}^{(i)})} \\ &= \frac{\pi_t(X_{0:t}^{(i)})}{q_t(X_{0:t}^{(i)}|X_{0:t-1}^{(i)})\pi_{t-1}(X_{0:t-1}^{(i)})}W_{t-1}^{(i)} \end{aligned} \tag{6}$$

and importance sampling estimate of $\mathbb{E}_{\pi_t}[\varphi_t(X_{0:t})]$ can be obtained using $\{W_t^{(i)}, X_{0:t}^{(i)}\}_{i=1}^{N}$.

However this approach fails as t becomes large. The weights tend to become concentrated on a few particles as the discrepancy between η_t and π_t becomes larger. Resampling techniques are applied such that, a new particle system $\{\bar{W}_t^{(i)}, \bar{X}_{0:t}^{(i)}\}_{i=1}^{M}$ is obtained with the property,

$$\mathbb{E}\Big[\sum_{i=1}^{M}\bar{W}_t^{(i)}\varphi_t(\bar{X}_{0:t}^{(i)})\Big] = \mathbb{E}\Big[\sum_{i=1}^{N}W_t^{(i)}\varphi_t(X_{0:t}^{(i)})\Big] \tag{7}$$

In practice, the resampling algorithm is usually chosen such that $M = N$ and $\bar{W}^{(i)} = 1/N$ for $i = 1,\ldots,N$. Resampling can be performed at each time t or adaptively based on some criteria of the discrepancy. One popular quantity used to monitor the discrepancy is *effective sample size* (ESS), introduced by Liu and Chen (1998), defined as

$$\text{ESS}_t = \frac{1}{\sum_{i=1}^{N}(W_t^{(i)})^2} \tag{8}$$

where $\{W_t^{(i)}\}_{i=1}^{N}$ are the normalized weights. And resampling can be performed when $\text{ESS} \leq \alpha N$ where $\alpha \in [0,1]$.

The common practice of resampling is to replicate particles with large weights and discard those with small weights. In other words, instead of generating a random sample $\{\bar{X}_{0:t}^{(i)}\}_{i=1}^{N}$ directly, a random sample of integers $\{R^{(i)}\}_{i=1}^{N}$ is generated, such that $R^{(i)} \geq 0$ for $i = 1,\ldots,N$ and $\sum_{i=1}^{N} R^{(i)} = N$. And each particle value $X_{0:t}^{(i)}$ is replicated for $R^{(i)}$ times in the new particle system. The distribution of $\{R^{(i)}\}_{i=1}^{N}$ shall fulfill the requirement of Equation (7). One such distribution is a multinomial distribution of size N and weights $(W_t^{(i)}, \dots, W_t^{(N)})$. See Douc, Cappé, and Moulines (2005) for some commonly used resampling algorithms.

2.2 SMC SAMPLERS

SMC samplers allow us to obtain, iteratively, collections of weighted samples from a sequence of distributions $\{\pi_t\}_{t\geq 0}$ over essentially any random variables on some spaces $\{E_t\}_{t\geq 0}$, by constructing a sequence of auxiliary distributions $\{\tilde{\pi}_t\}_{t\geq 0}$ on spaces of increasing dimensions, $\tilde{\pi}_t(x_{0:t}) = \pi_t(x_t)\prod_{s=0}^{t-1} L_s(x_{s+1}, x_s)$, where the sequence of Markov kernels $\{L_s\}_{s=0}^{t-1}$, termed backward kernels, is formally arbitrary but critically influences the estimator variance. See Del Moral, Doucet, and Jasra (2006b) for further details and guidance on the selection of these kernels.

Standard sequential importance sampling and resampling algorithms can then be applied to the sequence of synthetic distributions, $\{\tilde{\pi}_t\}_{t\geq 0}$. At time $t-1$, assume that a set of weighted particles $\{W_{t-1}^{(i)}, X_{0:t-1}^{(i)}\}_{i=1}^{N}$ approximating $\tilde{\pi}_{t-1}$ is available, then at time t, the path of each particle is extended with a Markov kernel say, $K_t(x_{t-1}, x_t)$ and the set of particles $\{X_{0:t}^{(i)}\}_{i=1}^{N}$ reach the distribution $\eta_t(X_{0:t}^{(i)}) = \eta_0(X_0^{(i)})\prod_{k=1}^{t} K_t(X_{t-1}^{(i)}, X_t^{(i)})$, where η_0 is the initial distribution of the particles. To correct the discrepancy between η_t and $\tilde{\pi}_t$, Equation (6) is applied and in this case,

$$W_t^{(i)} \propto \frac{\tilde{\pi}_t(X_{0:t}^{(i)})}{\eta_t(X_{0:t}^{(i)})} = \frac{\pi_t(X_t^{(i)})\prod_{s=0}^{t-1} L_s(X_{s+1}^{(i)}, X_s^{(i)})}{\eta_0(X_0^{(i)})\prod_{k=1}^{t} K_t(X_{t-1}^{(i)}, X_t^{(i)})} \propto \tilde{w}_t(X_{t-1}^{(i)}, X_t^{(i)})W_{t-1}^{(i)} \tag{9}$$

where $\tilde{w}_t$, termed the *incremental weights*, are calculated as,

$$\tilde{w}_t(X_{t-1}^{(i)}, X_t^{(i)}) = \frac{\pi_t(X_t^{(i)})L_{t-1}(X_t^{(i)}, X_{t-1}^{(i)})}{\pi_{t-1}(X_{t-1}^{(i)})K_t(X_{t-1}^{(i)}, X_t^{(i)})} \tag{10}$$

If π_t is only known up to a normalizing constant, say $\pi_t(x_t) = \gamma_t(x_t)/Z_t$, then we can use the *unnormalized* incremental weights

$$w_t(X_{t-1}^{(i)}, X_t^{(i)}) = \frac{\gamma_t(X_t^{(i)})L_{t-1}(X_t^{(i)}, X_{t-1}^{(i)})}{\gamma_{t-1}(X_{t-1}^{(i)})K_t(X_{t-1}^{(i)}, X_t^{(i)})} \tag{11}$$

for importance sampling. Further, with the previously *normalized* weights $\{W_{t-1}^{(i)}\}_{i=1}^{N}$, we can estimate the ratio of normalizing constant Z_t/Z_{t-1} by

$$\frac{\hat{Z}_t}{Z_{t-1}} = \sum_{i=1}^{N} W_{t-1}^{(i)} w_t(X_{t-1}^{(i)}, X_t^{(i)}) \tag{12}$$

Sequentially, the normalizing constant between initial distribution π_0 and some target π_T, $T \geq 1$ can be estimated. See Del Moral, Doucet, and Jasra (2006b) for details on calculating the incremental weights. In practice, when K_t is invariant to π_t, and an approximated suboptimal backward kernel

$$L_{t-1}(x_t, x_{t-1}) = \frac{\pi(x_{t-1})K_t(x_{t-1}, x_t)}{\pi_t(x_t)} \tag{13}$$

is used, the unnormalized incremental weights will be

$$w_t(X_{t-1}^{(i)}, X_t^{(i)}) = \frac{\gamma_t(X_{t-1}^{(i)})}{\gamma_{t-1}(X_{t-1}^{(i)})}. \tag{14}$$

2.3 OTHER SEQUENTIAL MONTE CARLO ALGORITHMS

Some other commonly used sequential Monte Carlo algorithms can be viewed as special cases of algorithms introduced above. The annealed importance sampling (AIS; Neal (2001)) can be viewed as SMC samplers without resampling.

Particle filters as seen in the physics and signal processing literature, can also be interpreted as the sequential importance sampling and resampling algorithms. See Doucet and Johansen (2011) for a review of this topic. A simple particle filter example is used in Section **??** to demonstrate basic features of the vSMC library.

3 USING THE vSMC LIBRARY

The library is hosted at GitHub. One can download the stable Releases or the development branch from the Git repository. This is a header only template C++ library. To install the library just move the contents of the `include` directory into a proper place, e.g., `/usr/local/include` on Unix-alike systems. Alternatively, one can use CMake (version 2.8.3 or later required).

```
cd /path_to_vSMC_source
mkdir build
cd build
cmake ..
make install
```

One may need administrator permissions to perform the last installation step, or change the destination using `-DCMAKE_INSTALL_PREFIX`

REFERENCES

Cappé, Olivier, Simon J. Godsill, and Eric Moulines (2007). “An overview of existing methods and recent advances in sequential Monte Carlo”. In: *Proceedings of the IEEE* 95.5, pp. 899–924.

Del Moral, Pierre, Arnaud Doucet, and Ajay Jasra (2006a). “Sequential Monte Carlo methods for Bayesian computation”. In: *Bayesian Statistics 8*. Oxford University Press,

— (2006b). “Sequential Monte Carlo samplers”. In: *Journal of Royal Statistical Society B* 68.3, pp. 411–436.

Douc, Randal, Olivier Cappé, and Eric Moulines (2005). “Comparison of resampling schemes for particle filtering”. In: *Proceedings of the 4th International Symposium on Imange and Signal Processing and Analysis*, pp. 1–6.

Doucet, Arnaud and Adam M. Johansen (2011). “A tutorial on particle filtering and smoothing: Fifteen years later”. In: *The Oxford Handbook of Non-linear Filtering*. Oxford University Press,

Johansen, Adam M. (2009). “SMCTC: sequential Monte Carlo in C++”. In: *Journal of Statistical Software* 30.6, pp. 1–41.

Lee, Anthony et al. (2010). “On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods”. In: *Journal of Computational and Graphical Statistics* 19.4, pp. 769–789.

Liu, Jun S. and Rong Chen (1998). “Sequential Monte Carlo methods for dynamic systems”. In: *Journal of the American Statistical Association* 93.443, pp. 1032–1044.

Neal, Radford M. (2001). “Annealed importance sampling”. In: *Statistics and Computing* 11.2, pp. 125–139.

Peters, Gareth W (2005). “Topics in sequential Monte Carlo samplers”. MA thesis.