Parallel Particle MCMC with Poisson Resampling

B. Miasojedow, W. Niemiro and M. Startek

July 7, 2017

Abstract

We introduce a new version of particle filter in which the number of “children” of a particle at a given time has a Poisson distribution. As a result, the number of particles is random and varies with time. An advantage of this scheme is that descendents of different particles can evolve independently. It makes easy to parallelize computations. We also show that the basic techniques of particle MCMC, namely particle independent Metropolis-Hastings, particle Gibbs Sampler and its version with backward sampling of ancestors, work under our Poisson resampling scheme. We prove that versions of these algorithms, suitably modified to our setup, preserve the target distribution on the space of trajectories of hidden Markov process.

Keywords: Sequential Monte Carlo, Particle Markov chain Monte Carlo, Parallel computations, Poisson distribution, Hidden Markov process, Pseudo-marginal, Independent Metropolis-Hastings Algorithm, Gibbs Sampler, Ancestor Sampling.

1 Introduction

Particle Filters (PF) and more general Sequential Monte Carlo methods (SMC) (Doucet et al., 2001) are powerful numerical methods routinely used to perform statistical inference. SMC inference technique require synchronization at resampling step and due to that fact parallelization of SMC is challenging. In the literature there exist several approaches to construct parallel SMC algorithms, for example Paige et al. (2014); Murray et al. (2016). In the current note we introduce a Poisson resampling scheme which could be easily parallelized. Our algorithm similarly to Paige et al. (2014) generates a branching process. The main difference from Paige et al. (2014) is that the Poisson resampling allows us to use our algorithm inside particle MCMC methods (Andrieu et al., 2010). Moreover, our framework allows us to perform ancestor sampling in the particle Gibbs algorithm in the spirit of Lindsten et al. (2014).
1 INTRODUCTION

Why Poisson resampling works

Let us first consider an importance sampling procedure with Poisson resampling. Let $\nu$ be a probability density on $\mathcal{X}$. The target density is

$$\pi(x) = \frac{\nu(x)w(x)}{z},$$

where $w$ is a weight (importance) function and $z = \int_{\mathcal{X}} \nu(x)w(x)dx$.

Consider the following sampling scheme. Draw $N \sim \text{Poisson}(\lambda)$. If $N = 0$ then do nothing and put $\hat{Z} = 0$. If $N > 0$ then draw independently $X_1, \ldots, X_N \sim \nu(\cdot)$. Put $\hat{Z} = \sum_{m=1}^{N} w(X_m)/\lambda$.

It is obvious that $\mathbb{E}\hat{Z} = z$. Choose $S \in \{1 : N\}$ with probability $P(S = s | N) = \frac{w(X_s)}{\sum_{m=1}^{N} w(X_m)}$.

We say that the joint probability distribution of all the random variables generated in such a way is the extended proposal. It is denoted by $\psi$ and given by

$$(1.1) \quad \psi(n, x_1, \ldots, x_n, s) = e^{-\lambda} \frac{\lambda^n}{n!} \prod_{m=1}^{n} \nu(x_m) \frac{w(x_s)}{\sum_{m=1}^{n} w(x_m)},$$

for $n > 0$ and $\psi(0) = e^{-\lambda}$. Note that $\psi$ is defined on the space $\{0\} \cup \bigcup_{n=1}^{\infty} \{n\} \times \mathcal{X}^n \times \{1 : n\}$.

Define the extended target distribution by

$$(1.2) \quad \phi(n, x_1, \ldots, x_n, s) = e^{-\lambda} \frac{\lambda^n}{n!} \prod_{m=1}^{n} \nu(x_m) \frac{w(x_s)}{\lambda z}$$

for $n = 1, 2, \ldots$ and $\phi(0) = 0$. Note that $\phi$ can be decomposed as follows:

$$(1.3) \quad \phi(n, x_1, \ldots, x_n, s) = \underbrace{\frac{\nu(x_s)w(x_s)}{z}}_{\text{marginal}} \cdot \frac{1}{n} e^{-\lambda} \frac{\lambda^{n-1}}{(n-1)!} \prod_{m\neq s} \nu(x_m).$$

Formula (1.3) shows that $\phi$ is properly normalized, the marginal distribution of $X = X_S$ is exactly $\pi(\cdot)$. The conditional distribution of all the remaining variables can be obtained in the following way. The number of the other particles, $N - 1$ has the Poisson distribution. Once $N - 1$ is selected, we assign $X$ label $S$ chosen uniformly at random from the set $\{1 : N\}$, then draw $N - 1$ samples from $\nu(\cdot)$ and assign them labels $\{1 : N\} \setminus \{S\}$.

If we start with $X \sim \pi(\cdot)$, then configuration $(N, X_1, \ldots, X_N, S)$ such that $X = X_S$, produced in the way described above has the extended target distribution. From formula (1.2) it is clear that the conditional distribution of $S$ given $N, X_1, \ldots, X_N$ is proportional to $w(X_S)$. 

If we select new \( S' \) from this distribution then \( X_{S'} \sim \pi(\cdot) \). Thus the Particle Gibbs Sampler (pGS) preserves the target.

To see that the Particle Independent Metropolis-Hastings (pIMH) also preserves the target, it is enough to note that

\[
\frac{\phi(n, x_1, \ldots, x_n, s)}{\psi(n, x_1, \ldots, x_n, s)} = \hat{Z} / z.
\]

## 2 Poisson Forest Particle Filter

The target distribution is defined on \( X = \prod_{t=1}^{k} \mathcal{X}_t \). Its density with respect to a fixed product measure is given by

\[
\pi(x_{1:k}) = \frac{1}{z} \nu(x_1) w_1(x_1) \prod_{t=1}^{k-1} K_t(x_t, x_{t+1}) w_{t+1}(x_{t+1}),
\]

where \( \nu \) is a probability distribution on \( \mathcal{X}_1 \), \( K_t \) is a Markov kernel from \( \mathcal{X}_t \) to \( \mathcal{X}_{t+1} \) and \( w_t : \mathcal{X}_t \to [0, \infty) \) is a weight (importance) function. (Usually, in state-space models, weights are equal to the likelihood of observed variables: \( w_t(x_t) = p(y_t, x_t) \).) The norming constant is denoted by \( z \). To simplify notation, we identify measures with their densities.

In words, our basic algorithm can be described quite simply. At moment \( t \), we have a random number of existing particles in the space \( \mathcal{X}_t \). Assume that particle \( X_i \) is assigned a weight \( W_i = w_t(X_i) \) and an intensity parameter \( \Lambda_i \) which may depend on the current configuration of particles (and even on the history of the process). Particle \( X_i \) gives birth to \( N_i \) “children”, where \( N_i \sim \text{Pois}(\Lambda_iW_i) \). Children are independently “moved” according to the probability distribution \( K_t(X_i, \cdot) \) to the space \( \mathcal{X}_{t+1} \). Note that some of the particles have no children. In the end we select one of the particles existing at moment \( k \), say \( X_s \), with probability proportional to \( W_s / C_{pa(s)} \), where \( C_{pa(s)} \) is the product of \( \Lambda_i \) over all ancestors \( i \) of \( s \) excluding \( s \) itself.

To formally define the algorithm we introduce suitable notations. The Particle Filter with Poisson Resampling produces a random structure \( \mathbb{F} = (\mathcal{I}, \mathcal{E}, \mathbf{X}, S) \).

- \((\mathcal{I}, \mathcal{E})\) is a directed graph with the set \( \mathcal{I} \) of nodes and set \( \mathcal{E} \) of edges (arrows). A generic element of \( \mathcal{I} \) is denoted by \( i \) and a generic element of \( \mathcal{E} \) by \( i \to j \). Graph \((\mathcal{I}, \mathcal{E})\) is acyclic and every its node has at most one parent (incoming edge), so it is a directed forest. If \( i \to j \) then we write \( i = pa(j) \) and \( j \in ch(i) \). For any \( i \in \mathcal{I} \) there is a unique ancestry line denoted by \( an(i) \). It is a sequence of nodes \( (a_1(i), \ldots, a_t(i)) \) such that \( a_t(i) = i, a_r(i) = pa(a_{r+1}(i)) \) for \( r \in \{1 : t - 1\} \) and \( pa(a_1(i)) = \emptyset \). The set of nodes is partitioned into \( k \) subsets: \( \mathcal{I} = \bigcup_{t=1}^{k} \mathcal{I}_t \), where \( \mathcal{I}_t \) is the set of nodes with exactly \( t \) ancestors (note that \( i \in an(i) \)). We interpret \( t \) as “time” and \( \mathcal{I}_t \) as \( t \)th “generation of
particles”. The set of arrows going from $i$ is $\text{ar}(i) = \{ i \rightarrow j : j \in \text{ch}(i) \}$. Write also $\mathcal{E}_t = \bigcup_{i \in \mathcal{I}_t} \text{ar}(i)$, so that $\mathcal{E} = \bigcup_{t=1}^{k-1} \mathcal{E}_t$.

- $\mathbf{X} = (X_i : i \in \mathcal{I})$ is a collection of random variables with values in $\bigcup_{t=1}^{k} \mathcal{X}_t$ (if $i \in \mathcal{I}_t$ then $X_i \in \mathcal{X}_t$ denotes location of particle $i$ at time $t$). Configuration of particles at time $t$ is $X_{\mathcal{I}_t} = (X_i : i \in \mathcal{I}_t)$. We also write $X_{\text{an}(i)} = (X_{\alpha_1(i)}, \ldots, X_{\alpha_t(i)})$.

- $S \in \mathcal{I}_k$ is a (random) node identifying a selected particle.

Let us say $\mathbf{F}$ is a Poisson Forest.

Apart from the weight functions $w_t$ and Markov kernels $K_t$, we consider intensity functions $\ell_t : \mathcal{I}_t \times \mathcal{X}_t^2 \to [0, \infty)$ for $t \in \{1 : k-1\}$. Collections of random variables $(W_i : i \in \mathcal{I}_t)$ and $(\Lambda_i : i \in \mathcal{I}_t)$ are defined by $W_i = w_t(X_i)$ and $\Lambda_i = \ell_t(i, X_i)$. The main ingredients of our basic algorithm are defined separately as two functions. Function $M_0(N_0)$ generates $N_0$ initial particles; function $M(i, N_i)$ generates $N_i$ children of particle $X_i$. Both these functions return the labels of the new particles (nodes of the graph), their locations and weights.

**Algorithm PFPF (Poisson Forest Particle Filter)**

```plaintext
{ Initialize: }
C_0 := \lambda_0
Sample N_0 \sim \text{Poiss}(\lambda_0)
(\mathcal{I}_1, X_{\mathcal{I}_1}, W_{\mathcal{I}_1}) := M_0(N_0) \quad \{ Create initial generation of particles \}
{ Main loop: }
for $t := 1$ to $k-1$ do
    $\mathcal{I}_{t+1} := \emptyset$; $\mathcal{E}_{t+1} := \emptyset$
    for all $i \in \mathcal{I}_t$ do
        $\Lambda_i := \ell_t(i, X_{\mathcal{I}_t})$
        $C_i := C_{\text{pa}(i)} \Lambda_i$
        Sample $N_i \sim \text{Poiss}(\Lambda_i W_i)$
        $(\text{ch}(i), \text{ar}(i), X_{\text{ch}(i)}, W_{\text{ch}(i)}) := M(i, N_i) \quad \{ Create children of $i$ \}$
        $\mathcal{I}_{t+1} := \mathcal{I}_{t+1} \cup \text{ch}(i)$ \quad \{ Add new nodes \}
        $\mathcal{E}_{t+1} := \mathcal{E}_{t+1} \cup \text{ar}(i)$ \quad \{ Add new edges \}
    end for
end for
if $\mathcal{I}_k \neq \emptyset$ then
    $\hat{Z} := \sum_{i \in \mathcal{I}_k} W_i / C_{\text{pa}(i)}$
    Select $S \in \mathcal{I}_k$ from the probability distribution $\mathbb{P}(S = s) \propto W_s / C_{\text{pa}(s)}$
else
    $\hat{Z} := 0$
end if
Output $\hat{Z}$, $X^*_{\mathcal{I}_k} := X_{\text{an}(S)}$
```
Functions $M_0(N_0)$ and $M(i, N_i)$ are defined as follows.

Function $M_0(N_0)$

Create the set $\mathcal{I}_1$ of root nodes, where $|\mathcal{I}_1| = N_0$ \{ Labels of initial particles \}

for all $i \in \mathcal{I}_1$ do

    Sample $X_i \sim \nu(\cdot)$ \{ Initialize particle $i$ \}
    $W_i := w_1(X_i)$ \{ Weigh particle $i$ \}

end for

Return $(\mathcal{I}_1, X_{\mathcal{I}_1}, W_{\mathcal{I}_1})$

Function $M(i, N_i)$

{ Assume that $i \in \mathcal{I}_t$ }

$\text{ar}(i) := \emptyset$

Create the set $\text{ch}(i)$ of nodes, where $|\text{ch}(i)| = N_i$ \{ Children of node $i$ \}

for all $j \in \text{ch}(i)$ do

    $\text{ar}(i) := \text{ar}(i) \cup \{i \rightarrow j\}$ \{ Add edge $i \rightarrow j$ \}
    Sample $X_j \sim K_t(X_i, \cdot)$ \{ Move particle from $X_i$ to $X_j$ \}
    $W_j := w_{t+1}(X_j)$ \{ Weigh particle $j$ \}

end for

Return $(\text{ch}(i), \text{ar}(i), X_{\text{ch}(i)}, W_{\text{ch}(i)})$

2.1 Remark (Homogeneity of the intensities). The algorithm PFPF slightly simplifies if we assume that $\lambda_t(i, X_{\mathcal{I}_t}) = \lambda_t(X_{\mathcal{I}_t})$. Then the intensity parameters are the same for all particles in $t$th generation, say $\Lambda_i = \Lambda^{(t)}$ for $i \in \mathcal{I}_t$. Consequently,

$$\hat{Z} = \frac{1}{C^{(k-1)}} \sum_{i \in \mathcal{I}_k} W_i,$$

where $C^{(k-1)} = \lambda_0 \prod_{t=1}^{k-1} \Lambda^{(t)}$ and

$$\mathbb{P}(S = s) \propto W_s.$$

This simplification will be needed in the procedure of parent sampling, in the algorithm PFGS to be defined later.

The probability distribution of configurations produced by the filter is illustrated by the following example.
2.2 EXAMPLE. The forest produced by PFPF might look as below:

\[
\begin{align*}
\text{\texttt{t}} & = 1 \quad \text{\texttt{t}} = 2 \quad \text{\texttt{t}} = 3 \\
1 & \rightarrow 11 \rightarrow 112 \\
& \downarrow \downarrow \quad \uparrow \uparrow \\
& 111 \\
\downarrow \downarrow \quad \uparrow \uparrow \\
& 113 \\
& \rightarrow 21 \rightarrow 211 \\
2 & \rightarrow 22 \\
& \rightarrow 23 \rightarrow 231 \\
& \rightarrow 232
\end{align*}
\]

(We use a special convention of labelling nodes with their full ancestry lines.)

We have \( N_0 = 2, N_1 = 1, N_2 = 3, N_{11} = 3, N_{21} = 1, N_{22} = 0, N_{23} = 2 \). Assume that \( k = 3 \) and the selected node is \( S = 231 \). Its ancestry line is indicated by double arrows. To simplify the notation, we drop subscripts \( t \) in functions \( \lambda_t, w_t \) and \( K_t \). The probability of sampling the forest in our example is

\[
\psi(\mathcal{I}, \mathcal{E}, \mathbf{x}, s) = \exp \left[ -\lambda_0 \frac{\lambda_0^2}{2!} \nu(x_1) \nu(x_2) \right] \\
\times \exp \left[ -\lambda_1 w(x_1) \right] \frac{[\lambda_1 w(x_1)]^1}{1!} K(x_1, x_{11}) \\
\times \exp \left[ -\lambda_2 w(x_2) \right] \frac{[\lambda_2 w(x_2)]^3}{3!} K(x_2, x_{21}) K(x_2, x_{22}) K(x_2, x_{23}) \\
\times \exp \left[ -\lambda_{11} w(x_{11}) \right] \frac{[\lambda_{11} w(x_{11})]^3}{3!} K(x_{11}, x_{111}) K(x_{11}, x_{112}) K(x_{11}, x_{113}) \\
\times \exp \left[ -\lambda_{21} w(x_{21}) \right] \frac{[\lambda_{21} w(x_{21})]^1}{1!} K(x_{21}, x_{211}) \\
\times \exp \left[ -\lambda_{22} w(x_{22}) \right] \\
\times \exp \left[ -\lambda_{23} w(x_{23}) \right] \frac{[\lambda_{23} w(x_{23})]^2}{2!} K(x_{23}, x_{231}) K(x_{23}, x_{232}) \\
\times \frac{w(x_{231})}{\lambda_0 \lambda_2 \lambda_{23}} \cdot \frac{1}{Z}
\]
where $\lambda_1 = \ell_1(1, x_1, x_2), \lambda_2 = \ell_1(2, x_1, x_2), \lambda_{11} = \ell_2(11, x_{11}, x_{21}, x_{22}, x_{23})$ and so forth. The estimator of the norming constant $z$ is given by

$$\hat{Z} = \frac{w(x_{111})}{\lambda_0 \lambda_1 \lambda_{11}} + \frac{w(x_{112})}{\lambda_0 \lambda_1 \lambda_{11}} + \frac{w(x_{113})}{\lambda_0 \lambda_1 \lambda_{11}} + \frac{w(x_{211})}{\lambda_0 \lambda_2 \lambda_{21}} + \frac{w(x_{231})}{\lambda_0 \lambda_2 \lambda_{23}} + \frac{w(x_{232})}{\lambda_0 \lambda_2 \lambda_{23}}.$$ 

We can therefore rewrite $\psi$ (the extended proposal) as

$$\psi(\mathcal{I}, \mathcal{E}, x, s) = \exp \left[ -\lambda_0 \frac{\nu(x)}{1!} \right]$$

$$\times \exp \left[ -\lambda_1 w(x_1) \right] \frac{[\lambda_1 w(x_1)]^1}{1!} K(x_1, x_{11})$$

$$\times \exp \left[ -\lambda_2 w(x_2) \right] \frac{[\lambda_2 w(x_2)]^2}{2!} K(x_2, x_{21}) K(x_2, x_{22})$$

$$\times \exp \left[ -\lambda_{11} w(x_{11}) \right] \frac{[\lambda_{11} w(x_{11})]^3}{3!} K(x_{11}, x_{111}) K(x_{11}, x_{112}) K(x_{11}, x_{113})$$

$$\times \exp \left[ -\lambda_{21} w(x_{21}) \right] \frac{[\lambda_{21} w(x_{21})]^1}{1!} K(x_{21}, x_{211})$$

$$\times \exp \left[ -\lambda_{22} w(x_{22}) \right]$$

$$\times \exp \left[ -\lambda_{23} w(x_{23}) \right] \frac{[\lambda_{23} w(x_{23})]^1}{1!} K(x_{23}, x_{232})$$

$$\times \frac{1}{2 \cdot 3 \cdot 2} \frac{\nu(x_2) w(x_2) K(x_2, x_{32}) w(x_{23}) K(x_{23}, x_{231}) w(x_{231})}{\hat{Z}}.$$ 

The extended target distribution obtains if we replace $\hat{Z}$ by $z$ in the above expression, i.e. put

$$\phi(\mathcal{I}, \mathcal{E}, x, s) = \psi(\mathcal{I}, \mathcal{E}, x, s) \frac{\hat{Z}}{z}.$$ 

The terms corresponding to the selected path (indicated in blue) become the properly normalized target distribution:

$$\frac{\nu(x_2) w(x_2) K(x_2, x_{32}) w(x_{23}) K(x_{23}, x_{231}) w(x_{231})}{z} = \pi(x_2, x_{23}, x_{231}).$$ 

The remaining terms describe the distribution of a PFPF conditional on the selected path (at each generation the number of children is selected as in PFPF but then increased by one and the selected path is immersed in the growing forest). Factor $2 \cdot 3 \cdot 2$ is the number of ways the selected path can be labelled. A formal definition of the conditional PFPF is given in the next section in the algorithm PFGS. \triangle
2.3 REMARK (Labels of nodes). The labels given to the nodes of the graph \((\mathcal{I}, \mathcal{E})\) are irrelevant to the behaviour of the algorithm. However, these labels are part of the structure \(\mathcal{F} = (\mathcal{I}, \mathcal{E}, \mathbf{X}, S)\) generated by PFPF and the formulas for the probability distributions in Example 2.2 take this into account. Strictly speaking, we are interested in the equivalence classes \([\mathcal{F}] = [\mathcal{I}, \mathcal{E}, \mathbf{X}, S]\), where two structures are equivalent if they differ from each other only by labelling of the nodes. (That is, if there is a one-to-one correspondence between the sets of nodes which preserves the set of arrows and the \(X_i\)s.) It is easy to see that there are \(q(\mathcal{F}) = \prod_{i \in I} |\text{ch}(i)|!\) structures equivalent to \(\mathcal{F}\). Therefore \(\psi([\mathcal{F}]) = q(\mathcal{F}) \psi(\mathcal{F})\) and similarly for \(\phi\) (in Example 2.2 we should omit all the factorials \(2!1!3!3!1!2!\)). In the sequel we will work with the equivalence classes and drop the factorials in the formulas.

3 Probability distributions

Now we proceed to the description of the extended proposal and the extended target distributions. The former is the probability distribution governing the variables involved in PFPF: \(\mathcal{I}, \mathcal{E}, \mathbf{X}, S\). The latter is the probability distribution which has the desired marginal \(\pi(x_{\text{an}(s)})\).

Generalizing Example 2.2 and taking into account Remark 2.3, we see that the extended proposal (probability distribution of an equivalence class) is given by

\[
\psi([\mathcal{I}, \mathcal{E}, \mathbf{x}, s]) = M_0 \cdot \prod_{i \in \mathcal{I} \setminus \mathcal{I}_k} M(i) \cdot \frac{w_k(x_i)/c_{\text{pa}(i)}}{\sum_{i \in \mathcal{I}_k} w_k(x_i)/c_{\text{pa}(i)}},
\]

where

\[
M_0 = \exp [-\lambda_0] \lambda_0^{\mathcal{I}_1} \prod_{i \in \mathcal{I}_1} \nu(x_i),
\]

\[
M(i) = \exp [-\lambda_i w_t(x_i)] \prod_{j \in \text{ch}(i)} K_t(x_i, x_j) \quad \text{for} \quad i \in \mathcal{I}_t,
\]

\[
c_j = \lambda_0 \prod_{i \in \text{an}(j)} \lambda_i, \quad \lambda_i = \ell(i, x_{\mathcal{I}_t}), \quad \text{for} \quad i \in \mathcal{I}_t.
\]

In (3.1) and everywhere else we use the convention that \(\prod_{i \in \emptyset} \ldots = 1\). The last term in (3.1) is equal to 1 (and \(s\) is undefined) if \(\mathcal{I}_k = \emptyset\).

The extended target is concentrated on configurations with \(\mathcal{I}_k \neq \emptyset\) and is given by

\[
\phi([\mathcal{I}, \mathcal{E}, \mathbf{x}, s]) = \psi([\mathcal{I}, \mathcal{E}, \mathbf{x}, s]) \frac{\tilde{Z}}{Z} = \pi(x_{\text{an}(s)}) \cdot \tilde{M}_0 \cdot \prod_{i \in \mathcal{I} \setminus \mathcal{I}_k} \tilde{M}(i),
\]
where
\[
\tilde{M}_0 = \exp \left[ -\lambda_0 \right] \lambda_0^{\left| I_1 \right| - 1} \prod_{i \in I_1, i \notin an(s)} \nu(x_i),
\]
\[
\tilde{M}(i) = \begin{cases} 
\exp \left[ -\lambda_i w_t(x_i) \right] \left[ \lambda_i w_t(x_i) \right]^{\left| ch(i) \right| - 1} \prod_{j \in ch(i), j \notin an(s)} K_t(x_i, x_j) & \text{if } i \in an(s); \\
M(i) & \text{otherwise.}
\end{cases}
\]
Recall that
\[
\hat{Z} = \sum_{i \in I_k} \frac{w_k(x_i)}{cpa(i)}.
\]

### 4 Poisson Forest MCMC

The two main Particle MCMC algorithms are *Particle Independent Metropolis-Hastings* and *Particle Gibbs Sampler*. Their versions with Poisson resampling are PFMH and PFGS defined below. Both are the recipes for simulating a Markov chain \( X(0), X(1), \ldots, X(n), \ldots \) with the stationary distribution \( \pi \) on the space \( \mathcal{X} \) of trajectories \( x_{1:k} \). The rules of transition from \( X = X(n) \) to \( X' = X(n+1) \) are the following.

One step of PFMH (Poisson Forest Metropolis-Hastings)

| Input \((X_{1:k}, \hat{Z})\)  \{ Output of the previous step \} |
| Run PFPF to obtain \((X_{1:k}^*, \hat{Z}^*)\)  \{ Proposal \} |
| Sample \( U \sim U(0,1) \) |
| if \( U < \hat{Z}^*/\hat{Z} \) then |
| \((X_{1:k}', \hat{Z}') := (X_{1:k}^*, \hat{Z}^*)\)  \{ Accept \} |
| else |
| \((X_{1:k}', \hat{Z}') := (X_{1:k}, \hat{Z})\)  \{ Reject \} |
| end if |
| Output \((X_{1:k}', \hat{Z}')\) |

Our Particle Gibbs Sampler, just as its classical counterpart, can include the additional step of parent sampling. However, we first describe the basic version (without parent sampling). In the following algorithm PFGS we slightly abuse the notation: \( X_t \) for \( t \in \{1 : k\} \) should not be confused with \( X_i \) for \( i \in \mathcal{I} \).
One step of PFGS (Poisson Forest Gibbs Sampler)

Input: $X_{1:k}$ \{ Output of the previous step; the filter is run conditionally on $X_{1:k}$ \}
Create the sequence of labels $S_{1:k}$ \{ For the conditioning path \}
for $t := 1$ to $k$ do
  $X_{S_t} := X_t$
  $W_{S_t} := w_t(X_t)$
end for

{ Initialize: }
$C_\emptyset := \lambda_0$
Sample $\tilde{N}_0 \sim \text{Poiss}(\lambda_0)$ \{ $\tilde{N}_0 = N_0 - 1$ \}
$(\tilde{I}_1, X_{\tilde{I}_1}, W_{\tilde{I}_1}) := M_0(\tilde{N}_0)$ \{ Create initial particles \}
$I_1 = \tilde{I}_1 \cup \{ S_1 \}$

{ Main loop: }
for $t := 1$ to $k - 1$ do
  $I_{t+1} := \{ S_t \}; \mathcal{E}_{t+1} := \emptyset$
  for all $i \in I_t$ do
    $\Lambda_i := \ell_t(i, X_{\tilde{I}_t})$
    $C_i := C_{\text{pa}(i)} \Lambda_i$
    Sample $\tilde{N}_i \sim \text{Poiss}(\Lambda_i W_i)$ \{ $\tilde{N}_i = N_i - 1$ if $i \in S_{1:k}$ and $\tilde{N}_i = N_i$ otherwise \}
    $(\text{ch}(i), \text{ar}(i), X_{\text{ch}(i)}, W_{\text{ch}(i)}) := M(i, \tilde{N}_i)$ \{ Create children of $i$ \}
    $I_{t+1} := I_{t+1} \cup \text{ch}(i)$ \{ Add new nodes \}
    $\mathcal{E}_{t+1} := \mathcal{E}_{t+1} \cup \text{ar}(i)$ \{ Add new edges \}
  end for
end for
for $t := 1$ to $k - 1$ do
  $\mathcal{E}_t := \mathcal{E}_t \cup \{ S_t \rightarrow S_{t+1} \}$ \{ Add edges along the conditioning path \}
end for
Select $S' \in I_k$ from the probability distribution $\mathbb{P}(S' = s') \propto W_{s'}/C_{\text{pa}(s')}$
{ Select new path }
Output $X_{1:k}^* := X_{\text{an}(s')}$

Although PFGS does preserve $\pi$, its mixing properties are poor because of the well-known phenomenon of path-degeneration (as for the classical Particle GS). A remedy is to additionally introduce sampling of parents.

In the following procedure PFPG, we assume that the intensity parameters satisfy $\Lambda_i = \Lambda^{(t)}$ for all $i \in I_t$, as in Remark 2.1. (Without this assumption the probability of sampling parents would be much more complicated.)
Algorithm PFGP (Poisson Forest Gibbs with Parent sampling)

The algorithm is the same as PFGS with the following exceptions:

- The intensity parameters are computed as $\Lambda_i := \ell_t(X_{I_t}) \{ \text{equal within } I_t \}$. It is unnecessary to compute $C_i$.
- The last part of the algorithm from ‘for $t := 1$ to $k - 1$’ is replaced by
  
  for $t = 1$ to $k - 1$ do
  
  Sample $I \in I_t$ with probability $P(I = i) \propto W_i K_t(X_i, X_{S_{t+1}})$
  
  $E_t := E_t \cup \{ i \rightarrow S_{t+1} \}$ \{ Sample parent of $S_{t+1}$ \}
  
  end for

Select $S' \in I_k$ from the probability distribution $P(S' = s') \propto W_{s'}$

\{ Select new path \}

Output $X^{*}_{1:k} := X_{an(S')}$

To illustrate the parent sampling scheme let us analyze the following example.

4.1 EXAMPLE. Consider the following two forests with $k = 2$.

\[ \begin{array}{ccc}
  t = 1 & t = 2 & t = 1 \quad t = 2 \\
  1 & A & 1 & A \\
  & B & & B \\
  2 & C & 2 & C \\
  & D & & D \\
\end{array} \]

Let us denote the left configuration by $(I, E_1, x)$ and the right one by $(I, E_2, x)$. Assume that $x = (x_1, x_2, x_A, x_B, x_C, x_D)$ is fixed and the same in both configurations. In what follows, we consider the equivalence classes (that is we identify configurations with different labelling of the nodes, see Remark 2.3). The extended target distribution (marginalized with respect to the selected final node $S$) for the two equivalence classes is

\[
\phi([I, E_1, x]) = \exp \left[ -\lambda_0 \lambda^{\gamma}(x_1)x_2 \right] \\
\times \exp \left[ -\lambda^{(1)} w(x_1) \right] \left[ \lambda^{(1)} w(x_1) \right] K(x_1, x_A)K(x_1, x_B) \\
\times \exp \left[ -\lambda^{(1)} w(x_2) \right] \left[ \lambda^{(1)} w(x_2) \right] K(x_2, x_C)K(x_2, x_D) \\
\times \hat{Z} \frac{z}{Z},
\]
and
\[
\phi([I, E_2, x]) = \exp \left[ -\lambda_0 \lambda_2 \nu(x_1) \nu(x_2) \right]
\times \exp \left[ -\lambda_1 w(x_1) \right] [\lambda_1 w(x_1)] K(x_1, x_A)
\times \exp \left[ -\lambda_1 w(x_2) \right] [\lambda_1 w(x_2)]^3 K(x_2, x_B) K(x_2, x_C) K(x_2, x_D)
\times \frac{\hat{Z}}{z}
\]

Note that
\[
\hat{Z} = \frac{w(x_A)}{\lambda_0 \lambda(1)} + \frac{w(x_B)}{\lambda_0 \lambda(1)} + \frac{w(x_C)}{\lambda_0 \lambda(1)} + \frac{w(x_D)}{\lambda_0 \lambda(1)}
\]
is the same for both configurations. (If \(\lambda_1\) were not equal to \(\lambda_2\) then it would not be true.)

The probabilities of the two configurations differ only by terms indicated in blue in the middle lines in the above formulas. If we consider the configuration \((I, E_-, x)\) in which the arrow coming to \(B\) is removed then it is clear that
\[
\phi([I, E_i, x]) | \phi([I, E_-, x]) = \frac{w(x_i) K(x_i, x_B)}{w(x_1) K(x_1, x_B) + w(x_2) K(x_2, x_B)}, \quad (i = 1, 2).
\]
The choice between \(E_1\) and \(E_2\) is equivalent to sampling a parent node for \(B\), with probabilities \(P(i \to B) \propto w(x_i) K(x_i, x_B)\) (conditional on \([I, E_-, x]\)). \(\triangle\)

5 Main results

5.1 Proposition. Let \(f : \mathcal{X} \to \mathbb{R}\) be a nonnegative function (recall that \(\mathcal{X} = \prod_{t=1}^{k} \mathcal{X}_t\) is the space of trajectories \(x_{1:k}\)). If the structure \([I, E, X]\) is produced by PFPF then
\[
\hat{z} \pi(f) = \begin{cases} 
\sum_{i \in I_k} \frac{W_i}{C_{pa(i)}} f(X_{an(i)}) & \text{if } I_k \neq \emptyset; \\
0 & \text{if } I_k = \emptyset
\end{cases}
\]
is an unbiased estimator of \(z \pi(f) = z \mathbb{E}_\pi f(X_{1:k})\). In particular, \(\hat{Z}\) is an unbiased estimator of \(z\).

Proof. By (3.2), if \([I, E, X, S] \sim \phi\) then the marginal distribution of \(X_{an(S)}\) is \(\pi\). Therefore \(\mathbb{E}_\phi f(X_{an(S)}) = \pi(f)\). Again using (3.2), we see that
\[
\psi[I, E, x] \frac{W_s}{C_{pa(s)}} = \hat{Z} \psi[I, E, x, s] = z \phi[I, E, x, s],
\]
where $\psi[I, E, x]$ is the extended proposal with $s$ marginalized out and $(W_s/C_{pa(s)})/\hat{Z} = \mathbb{P}_\psi[S = s|I, E, x]$. Now it is enough to multiply both sides of the last display by $f(x_{\text{an}(s)})$ and sum over $x$ and $s \in \mathcal{I}_k$ to obtain the result. Unbiasedness of $\hat{Z}$ follows if we put $f \equiv 1$.

5.2 Theorem. Algorithms PFMH, PFGS and PFGP generate Markov chains having the equilibrium distribution equal to the target $\pi$.

Proof. The line of argument is almost the same as for the classical pMCMC algorithms with multinomial resampling. The crucial point is equation (3.2).

For PFMH, we use equation (3.2) to infer that

$$\frac{\hat{z}^*}{\hat{z}} = \frac{\phi[I^*, E^*, x^*, s^*] \psi[I, E, x, s]}{\phi[I, E, x, s] \psi[I^*, E^*, x^*, s^*]},$$

where $\hat{z}^*, I^*, E^*, x^*, s^*$ are new values produced by running PFIMH, while $\hat{z}, I, E, x, s$ are values from the previous step. It follows that this algorithm is a proper Metropolis-Hastings procedure with the proposal distribution $\psi[I, E, x, s]$ and the target $\phi[I, E, x, s]$ on the space of configurations $[I, E, x, s]$. The second equation in (3.2) shows that the distribution $\phi$ preserved by PFMH has the right marginal distribution $\pi(x_{\text{an}(s)})$.

For PFGS, note that factor $\tilde{M}_0 \prod_{i \in \mathcal{I} \setminus \mathcal{I}_k} \tilde{M}(i)$ in equation (3.2) is the conditional distribution of $[I, E, X]$ given $X_{1:k}$. By running PFPG we sample a configuration with this conditional distribution. If, at the input, $X_{1:k} \sim \pi$ then $[I, E, X]$ obtained by PFPG has the distribution $\phi$ marginalized with respect to $S$. Consequently, $[I, E, X, S'] \sim \phi$ with the marginal $X_{\text{an}(S')} \sim \pi$ at the output.

Finally, for PFGP, it is enough to note that the parent sampling is exactly generation of $E$ from the conditional distribution $\phi[E|I, E_{-}, X]$, where $E_{-}$ is the set of edges with one arrow removed. Repeating this sampling for arbitrarily chosen arrows preserves $\phi$.

6 Concluding Remarks

The main motivation for this work is to construct algorithms in which computations are performed in a parallel way. In principle, the structure of PFPF, PFMH and PFGS can be made perfectly parallel. The descendants of every node $i \in \mathcal{I}_t$ can evolve completely independently of other nodes in $\mathcal{I}_t$, if every parameter $\Lambda_i$ depends only on $X_i$ (if we set $\ell_t(i, \mathcal{T}_i) = \ell_t(X_i)$). However, this scenario is rather unrealistic. The problem is with controlling the number of particles. It is equally undesirable to allow for an uncontrolled increase and for a rapid decrease (or even extinction) of the population. An obvious way to
stabilize the number of particles is to choose

\[
\ell_t(i, X_{I_t}) = \frac{\lambda_0}{\sum_{j \in I_t} w_t(X_j)},
\]

because then \(\sum_{i \in I_t} \Lambda_i W_i = \lambda_0\) and consequently \(\mathbb{E}|I_t| = \lambda_0\). Let us also note that (6.1) is compatible with PFGP (satisfies the assumption required for efficient ancestor sampling). Unfortunately, (6.1) is the opposite extreme compared to \(\ell_t(i, X_{I_t}) = \ell(X_i)\) and destroys the parallel structure of computations.

Some special properties of the Poisson distribution offer a possible way to overcome these difficulties and efficiently parallelize computations. Well-known techniques of “thinning” and “superposition” can be used in sampling the Poisson forest \(\mathbb{F}\). It is possible to use some preliminary estimates of \(\sum_{j \in I_t} w_t(X_j)\) to compute “tentative” values of \(\Lambda_i\)s at every time \(t\). Then, in the next stage, the forest can be adjusted by sampling additional children and their descendants (superposition) or removing some children and their descendants (thinning).

References

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, jun 2010. doi: 10.1111/j.1467-9868.2009.00736.x.

Arnaud Doucet, Nando Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer New York, 2001. doi: 10.1007/978-1-4757-3437-9.

Fredrik Lindsten, Michael I. Jordan, and Thomas B. Schön. Particle gibbs with ancestor sampling. *Journal of Machine Learning Research*, 15:2145–2184, 2014. URL http://jmlr.org/papers/v15/lindsten14a.html.

Lawrence M. Murray, Anthony Lee, and Pierre E. Jacob. Parallel resampling in the particle filter. *Journal of Computational and Graphical Statistics*, 25(3):789–805, jul 2016. doi: 10.1080/10618600.2015.1062015.

Brooks Paige, Frank Wood, Arnaud Doucet, and Yee Whye Teh. Asynchronous anytime sequential monte carlo. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3410–3418. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5450-asynchronous-anytime-sequential-monte-carlo.pdf.