An Adaptive Sequential Monte Carlo Method for Approximate Bayesian Computation

Pierre Del Moral
Centre INRIA Bordeaux Sud-Ouest
& Institut de Mathématiques,
Université Bordeaux I,
33405 Talence cedex, France.
Email: Pierre.Del-Moral@inria.fr

Arnaud Doucet
The Institute of Statistical Mathematics,
4-6-7 Minami-Azabu, Minato-ku,
Tokyo 106-8569, Japan.
Email: Arnaud@ism.ac.jp

Ajay Jasra
Department of Mathematics,
Imperial College London,
180 Queens Gate,
London, SW7 2AZ, UK
Email: Ajay.Jasra@imperial.ac.uk

29th December 2008

Abstract

Approximate Bayesian computation (ABC) is a popular approach to address inference problems where the likelihood function is intractable, or expensive to calculate. To improve over Markov chain Monte Carlo (MCMC) implementations of ABC, the use of sequential Monte Carlo (SMC) methods has recently been suggested. Effective SMC algorithms that are currently available for ABC have a computational complexity that is quadratic in the number of Monte Carlo samples [4, 17, 19, 21] and require the careful choice of simulation parameters. In this article an adaptive SMC algorithm is proposed which admits a computational complexity that is linear in the number of samples and determines on-the-fly the simulation parameters. We demonstrate our algorithm on a toy example and a population genetics example.

Keywords: Approximate Bayesian computation, Markov chain Monte Carlo, sequential Monte Carlo.

1 Introduction

1.1 Background

Assume we are given a Bayesian model where $\pi(\theta)$ denotes the prior density of the parameter of interest $\theta \in \Theta$ and $f(y|\theta)$ is the likelihood of data $y \in \mathcal{D}$. It is of interest to compute expectations with respect to the resulting posterior density $\pi(\theta|\, y)$. If the likelihood term $f(y|\theta)$ is expensive or impossible to calculate, it is difficult to use standard computational tools, such as Markov chain Monte Carlo (MCMC), to sample from $\pi(\theta|\, y)$. ABC is an alternative to such techniques that only requires being able to sample from $f(\cdot|\theta)$. ABC seeks to draw inference from the following modified posterior density on $\Theta \times \mathcal{D}$

$$\pi_\epsilon(\theta, x|y) = \frac{\pi(\theta) f(x|\theta) \mathbb{I}_{A_{\epsilon,y}}(x)}{\int_{A_{\epsilon,y} \times \Theta} \pi(\theta) f(x|\theta) dx d\theta} \tag{1}$$

with $\epsilon > 0$ a tolerance level, $\mathbb{I}_B(\cdot)$ the indicator function of a given set B, $A_{\epsilon,y} = \{z \in \mathcal{D} : \rho(\eta(z), \eta(y)) < \epsilon\}$ where $\eta : \mathcal{D} \to \mathcal{S}$ represents some summary statistics and $\rho : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^+$ a distance metric. The idea behind ABC is that the summary statistics coupled with a small tolerance should provide a good approximation of the likelihood. Hence it is expected that $\pi_\epsilon(\theta|y) \approx \pi(\theta|y)$ ([3]).

Rejection sampling was originally proposed to sample from $\pi_\epsilon(\theta, x|y)$. However, the acceptance probability can be very small: several improvements over this algorithm have been developed. It was shown in [15] that it is possible to construct MCMC methods to sample from $\pi_\epsilon(\theta, x|y)$, which only require being able to sample from $f(x|\theta)$. These MCMC schemes can still perform poorly if the tolerance level ϵ is small. Consequently various sequential Monte Carlo (SMC) methods have been recently constructed as an alternative to MCMC methods [4, 17, 18, 21]. The key idea is to decompose the difficult problem of sampling from $\pi_\epsilon(\theta, x|y)$ into a series of simpler subproblems. The algorithm begins at time 0 sampling from $\pi_{\epsilon_0}(\theta, x|y)$, with ϵ_0 large, then simulating from an increasingly difficult sequence of target distributions $\pi_{\epsilon_n}(\theta, x|y)$, that is $\epsilon_n < \epsilon_{n-1}$, at subsequent time steps $n = 1, ..., T$. In other words the tolerance level is decreased until it reaches ϵ. These distributions are approximated by a large number of random samples named particles which are propagated over time using a combination of importance sampling (IS) and resampling.

In the ABC context, [18] used the SMC samplers methodology developed in [8] coupled with a partial rejection proposal (PRC). Some concerns have been raised about this algorithm [4]; this debate is not contributed to. It is just mentioned that [4, 19, 21] developed methods to improve the performance of the algorithm in [18] by using an approximation of the 'optimal' backward kernel in [8, Section 2.4]. This leads to algorithms of computational complexity that are quadratic in the number of particles and still requires a careful determination of the sequence of tolerance levels. Indeed, if the tolerance levels decrease too fast then the algorithm can perform poorly whereas if they decrease too slowly then the algorithm will be too computationally intensive.

1.2 Contributions and Organization of the Article

In this article an original adaptive SMC method for ABC is developed. In comparison to previous work, our algorithm has the following features:

- a computational complexity that is linear in the number of particles.
- it determines, in an automatic fashion, the sequence of tolerance levels to be used.
- it determines, in an automatic fashion, the parameters of some proposals.

Note that [4] adapts the parameters of proposal densities but not the tolerance levels.

The structure of this article is as follows. In Section 2 we outline the SMC sampler approach of [8] and discuss its application in an ABC context. In Section 3 an original adaptive SMC scheme for ABC is introduced. In Section 4 the performance of this algorithm is investigated on a toy example and a population genetics example. We discuss various extensions in Section 5.

2 Sequential Monte Carlo Samplers for Approximate Bayesian Computation

2.1 Sequential Monte Carlo Samplers

The SMC sampler methodology is a generic approach to approximately simulate from a sequence of related probability distributions $\{\pi_n\}_{0\leq n\leq T}$ defined upon a common measurable space $(E, \mathcal{E})$ [8]. At time 0, the distribution π_0 is selected such that it is easy to approximate it using IS. The particles are then moved, from time $n-1$ to time n, by using a Markov kernel K_n. As the resulting marginal distribution at time n is typically not available, IS cannot be used, directly, to correct for the discrepancy between this distribution and the target π_n. To bypass this problem, a sequence of extended probability distributions $\{\widetilde{\pi}_n\}_{0\leq n\leq T}$ are introduced on state-spaces of increasing dimension $(E^{n+1}, \mathcal{E}^{\otimes n+1})$ admitting $\{\pi_n\}_{0\leq n\leq T}$ as marginals; see [8] for details. More specifically, the following sequence of auxiliary densities is used

$$\widetilde{\pi}_n(z_{0:n}) = \pi_n(z_n)\prod_{j=0}^{n-1} L_j(z_{j+1}, z_j) \tag{2}$$

where $z_{0:n} := (z_0, \ldots, z_n)$, $\{L_n\}_{0\leq n\leq T-1}$ are a sequence of Markov kernels that act backward in time and are termed backward Markov kernels. It is clear from Eq. (2) that $\{\widetilde{\pi}_n\}$ admit $\{\pi_n\}$ as marginals.

The algorithm proceeds as follows. Note that $\delta_x(\cdot)$ is the Dirac measure.

- Step 0. Set $n = 0$; for $i = 1, \ldots, N$ sample $Z_0^{(i)} \sim \eta_0$ and compute $W_0^{(i)} \propto \pi_0\left(Z_0^{(i)}\right) / \eta_0\left(Z_0^{(i)}\right)$, $\sum_{j=1}^{N} W_0^{(j)} = 1$.
- Step 1. If $\mathrm{ESS}\left(\left\{W_n^{(i)}\right\}\right) < N_T$ then resample N particles from

$$\widehat{\pi}_n\left(dz\right) = \sum_{i=1}^{N} W_n^{(i)} \delta_{Z_n^{(i)}}\left(dz\right) \tag{3}$$

  also denoted abusively $\left\{Z_n^{(i)}\right\}$ and set $W_n^{(i)} = \frac{1}{N}$. Set $n = n + 1$, if $n = T + 1$ stop.
- Step 2. For $i = 1, \ldots, N$, sample $Z_n^{(i)} \sim K_n\left(Z_{n-1}^{(i)}, \cdot\right)$, compute

$$W_n^{(i)} \propto W_{n-1}^{(i)} \frac{\pi_n\left(Z_n^{(i)}\right) L_{n-1}\left(Z_{n-1}^{(i)}, Z_n^{(i)}\right)}{\pi_{n-1}\left(Z_{n-1}^{(i)}\right) K_n\left(Z_{n-1}^{(i)}, Z_n^{(i)}\right)} \tag{4}$$

  and return to Step 1.

The resampling step is implemented using the systematic resampling scheme [13] and only performed when the accuracy of the estimator is poor. Practically, this is usually assessed by looking at the variability of the weights using the so-called Effective Sample Size (ESS) criterion [14, pp. 35-36] given at time n by

$$\mathrm{ESS}\left(\left\{W_n^{(i)}\right\}\right) = \left(\sum_{i=1}^{N} \left(W_n^{(i)}\right)^2\right)^{-1}.$$

Its interpretation is that inference based on the N weighted samples is approximately equivalent to inference based on $\mathrm{ESS}\left(\left\{W_n^{(i)}\right\}\right)$ perfect samples from π_n. The ESS takes values between 1 and N and resampling only occurs when it is below a threshold N_T. Although the ESS is not a perfect measure, it does provide an idea of the behaviour of the algorithm - see [5] for some discussion on this.

2.2 Algorithm Settings for ABC

In the context of ABC, it is of interest to sample from a fixed target distribution $\pi_\epsilon\left(\theta|y\right)$ given by the marginal in θ of $\pi_\epsilon\left(\theta, x|y\right)$ in (1). As $\pi_\epsilon\left(\theta|y\right)$ is unknown, even up to a normalizing constant, SMC samplers techniques cannot be applied directly. Therefore, it is necessary to sample from the sequence of target distributions $\pi_n\left(z\right) = \pi_{\epsilon_n}(\theta, x|y)$ such that $\epsilon_0 > \epsilon_1 > \cdots > \epsilon_T = \epsilon$. It should be noted that it is also possible to use SMC to sample from the sequence of targets

$$\pi_{\epsilon_n}(\theta, x_{1:M}|y) \propto \left(\frac{1}{M}\sum_{k=1}^{M} \mathbb{I}_{A_{\epsilon_n,y}}(x_k)\right)\left(\prod_{k=1}^{M} f(x_k|\theta)\right)\pi(\theta) \tag{5}$$

for any integer $M \in \mathbb{N}$ [2]. This sequence admits the same marginal in θ for any M. Although it is more expensive to sample from $\pi_{\epsilon_n}(\theta, x_{1:M}|y)$ than $\pi_{\epsilon_n}(\theta, x|y)$ when $M > 1$, this has important advantages as discussed in Section 3.2 and illustrated in Section 4.

The performance of SMC samplers depends heavily upon the selection of an appropriate sequence $\{\epsilon_n\}$, the transition kernels $\{K_n\}$ and the backward transition kernels $\{L_n\}$. Assuming $\{\epsilon_n\}$ is fixed for the time being, it is recommended in [8] to use, for K_n, an MCMC kernel of invariant density π_n. This is the approach adopted later on, using a slightly improved version of the MCMC algorithm in [15]. Once K_n has been selected, the backward Markov kernel L_{n-1} is taken as

$$L_{n-1}(z, z') = \frac{\pi_n(z')K_n(z', z)}{\pi_n(z)}.$$

Alternative kernels yielding lower variance weights $\left\{W_n^{(i)}\right\}$ are given in [8] but they are not always applicable in this context. Note that this choice of backward kernels was implicitly made in [5, 11] and explicitly in related algorithms [6, 16] where no resampling step is used.

For such a selection of MCMC kernel K_n and reversal backward kernel L_{n-1}, it can be checked for π_n of the form given in Eq. (5), that Eq. (4) becomes

$$W_n^{(i)} \propto W_{n-1}^{(i)} \frac{\pi_n\left(Z_{n-1}^{(i)}\right)}{\pi_{n-1}\left(Z_{n-1}^{(i)}\right)} \propto W_{n-1}^{(i)} \frac{\sum_{k=1}^{M} \mathbb{I}_{A_{\epsilon_n, y}}\left(X_{k,n-1}^{(i)}\right)}{\sum_{k=1}^{M} \mathbb{I}_{A_{\epsilon_{n-1}, y}}\left(X_{k,n-1}^{(i)}\right)}. \tag{6}$$

In this very specific case, it is thus clear that if $M = 1$, $\eta_0 = \pi_0$ then either $W_n^{(i)} \propto 1$ or $W_n^{(i)} = 0$ and thus $\mathrm{ESS}\left(\left\{W_n^{(i)}\right\}\right)$ is directly proportional to the number of 'alive' particles at time $n-1$, that is to the number of particles with strictly positive weights $W_n^{(i)}$. It is also worth noticing that in this case $W_n^{(i)}$ is independent of $\left\{Z_n^{(i)}\right\}$. This allows us to swap the order of the sampling and resampling steps; see [8, Remark 1, p. 418]. We will also exploit this property in the next Section to obtain an adaptive method.

As $\epsilon_n < \epsilon_{n-1}$, it is typically the case that there is a non-null proportion of particles that have zero weights. This emphasizes the importance of selecting an appropriate sequence of tolerance levels. Indeed, if this sequence decreases too slowly then, with high probability, $W_n^{(i)} = W_{n-1}^{(i)}$ and the algorithm will move too slowly towards the target $\pi_\epsilon(\theta, x_{1:M}|y)$. Conversely, if the $\{\epsilon_n\}$ decrease too quickly, then, with high probability, all the weights $W_n^{(i)}$ can equal zero; hence the SMC sampler approximation would have collapsed. To prevent such a collapse, the algorithms in [17] and [18] targeting $\pi_{\epsilon_n}(\theta, x|y)$ generate particles $Z_n^{(i)} = \left(\theta_n^{(i)}, X_n^{(i)}\right)$ in regions such that $\rho(\eta\left(X_n^{(i)}\right), \eta(y)) < \epsilon_n$. The transition kernel K_n they use is not an MCMC kernel of invariant distribution $\pi_{\epsilon_n}(\theta, x|y)$ and this makes the selection of an associated backward kernel, to ensure that the variance of $W_n^{(i)}$ remains reasonable, more difficult. As mentioned in the introduction, effective SMC algorithms proposed to bypass this problem have a computational complexity that is quadratic in N [4, 17, 19, 21].

3 An Adaptive Sequential Monte Carlo Sampler for Approximate Bayesian Computation

In this section, a simple adaptive SMC algorithm is proposed which:

- relies on MCMC kernels to move the particles around the space,
- admits a computational complexity that is linear in N,
- automatically determines the sequence of tolerance levels so as to prevent the collapse of the SMC approximation.

3.1 An Adaptive Schedule for Tolerance Levels

Our method for selecting the tolerance level ϵ_n adaptively is based on the key remark that the expression (6) for the weights $\left\{W_n^{(i)}\right\}$ does not depend on $\left\{Z_n^{(i)}\right\} = \left\{\left(\theta_n^{(i)}, X_{1:M,n}^{(i)}\right)\right\}$; see [8, Section 3.5] for a detailed discussion. We aim to control the proportion of alive particles which is given by

$$\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n\right) := \frac{\sum_{i=1}^{N} \mathbb{I}_{\{W_n^{(i)}>0\}}}{N}$$

where $W_n^{(i)}$ given in Eq. (6) depends on ϵ_n. This criterion is equivalent to the ESS when $M = 1$. However for $M > 1$, the ESS is not an increasing function of ϵ_n contrary to this criterion and it is thus more difficult to interpret than $\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n\right)$. The proportion of alive particles is also intuitively a sensible measure of 'quality' of our SMC approximation. The tolerance level ϵ_n is selected to make sure that the proportion of alive particles is equal to a given percentage of the current value

$$\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n\right) = \alpha \mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right) \tag{7}$$

for $\alpha \in (0, 1)$. In practice, bisection is used to compute the root of (7). Hence, by construction, the SMC approximation can be prevented from collapsing. The parameter α is a 'quality' index for the resulting SMC approximation of the target. If $\alpha \approx 1$ then we will move slowly towards the target but the SMC approximation will be very good. However, if $\alpha \approx 0$ then we can move very quickly towards the target but the resulting SMC approximation will be unreliable. Finally, we resample the particle system if $\mathrm{ESS}\left(\left\{W_n^{(i)}\right\}\right) < N_T$.

In some cases, it may be the case that the choice of summary statistics leads to a distance metric ρ taking only discrete values; say integer values as in the population genetics example discussed in Section 4. In this case, Eq. (7) does not typically admit a solution and it is sensible to select tolerance levels taking the same values as the distance metric. Controlling the PA in these settings is more difficult as a discrete reduction in ϵ_n could lead to large

drops in the PA. Therefore, the possibility that ϵ_n does not fall at all, is introduced; let m_n be a counter of how long ϵ_n has not decreased. Our objective is relaxed, to ensure that

$$\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n\right) \in \left(\nu_{m_n}, \mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right)\right)$$

for some decreasing, in m_n, $\nu_{m_n} < \mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right)$.

We suggest the following strategy. Let ϵ_n^* be a dummy variable, that is decreased from ϵ_{n-1}, until

$$\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n^*\right) < \alpha \mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right). \tag{8}$$

Let $\nu_{m_n} \in \left(0, \alpha\mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right)\right)$. If $\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n^*\right) > \nu_{m_n}$ then let $\epsilon_n = \epsilon_n^*$ and $m_{n+1} = 0$. Otherwise let $\epsilon_n = \epsilon_{n-1}$, $m_{n+1} = m_n + 1$. Finally,

$$\nu_{m_{n+1}} = \eta \vee [\tau - \tau'(m_{n+1}+1)]\ \mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right)$$

where $\tau > \tau' \in (0,1)$ is such that $[\tau - \tau'(m_{n+1}+1)] < \alpha$ for any $m_{n+1} \geq 0$. The ν_{m_n} falls with m_n, but is never allowed to drop below $\eta \in (0,1)$. If we have $m_{n+1} = S$, the algorithm is terminated**.** The role of the parameters τ and τ' are quite clear; if they are set to allow $\left(\nu_{m_n}, \mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\}, \epsilon_{n-1}\right)\right)$ to be large then the algorithm will move forward quickly and vice-versa. In some cases, the PA can fall in a non-regular manner, especially when we cannot afford M to be large. Thus it could be useful to control the PA not only through $\{\epsilon_n\}$, but also the resampling mechanism: it might be preferable to resample whenever $\mathrm{PA}\left(\left\{W_n^{(i)}\right\}, \epsilon_n\right) < \beta$ where $\beta \in (0,1)$.

3.2 Adaptive MCMC Kernels

At each time our algorithm applies an MCMC kernel $K_n\left((\theta, x_{1:M}), (\theta', x'_{1:M})\right)$ of invariant density $\pi_{\epsilon_n}(\theta, x_{1:M}|y)$. A slightly modified version of the ABC-MCMC scheme of [15] can be used to achieve this. Given $Z = (\theta, X_{1:M})$, with $\sum_{k=1}^M \mathbb{I}_{A_{\epsilon_n,y}}(X_k) \geq 1$ then $(\theta^*, X^*_{1:M})$ are generated according to a proposal

$$q_n(\theta, \theta^*) \prod_{k=1}^M f\left(x_k^* | \theta^*\right).$$

This candidate is accepted with acceptance probability given by the Metropolis-Hastings (MH) ratio

$$1 \wedge \frac{\pi_{\epsilon_n}(\theta^*, X^*_{1:M}|y)}{\pi_{\epsilon_n}(\theta, X_{1:M}|y)} \frac{q_n(\theta^*, \theta)}{q_n(\theta, \theta^*)} \prod_{k=1}^M \frac{f\left(X_k | \theta\right)}{f\left(X_k^* | \theta^*\right)} = 1 \wedge \frac{\sum_{k=1}^M \mathbb{I}_{A_{\epsilon_n,y}}(X_k^*)}{\sum_{k=1}^M \mathbb{I}_{A_{\epsilon_n,y}}(X_k)} \frac{q_n(\theta^*, \theta)}{q_n(\theta, \theta^*)}.$$

This expression outlines the benefit of sampling M variables. We reduce the variance of the MH acceptance ratio as M increases. In the limiting case, i.e. $M \to \infty$, we have

$\frac{1}{M}\sum_{k=1}^{M}\mathbb{I}_{A_{\epsilon_n,y}}(X_k) \rightarrow \int f(x|\theta)\,\mathbb{I}_{A_{\epsilon_n,y}}(x)dx$ and our algorithm is similar to a 'marginal' MCMC algorithm where X has been integrated out analytically; see [2] for further discussion.

In this framework, we can adaptively determine the parameters of the proposal $q_n(\theta,\theta^*)$ based on the previous approximation of the target π_{n-1}. Contrary to adaptive MCMC methods [1], no stringent condition is required to ensure the validity of the algorithm as the MCMC kernel is only used to build a sensible importance distribution. In practice, the variance of θ under $\pi_{\epsilon_{n-1}}(\theta|y)$ is approximated at time $n-1$ using the SMC approximation. The resulting variance is used in the proposal density of the MCMC algorithm at time n, i.e. through a normal random walk proposal. Many other possible adaptation schemes are also possible.

3.3 An Adaptive Sequential Monte Carlo Method

Our adaptive SMC method for ABC proceeds as follows. We use $\epsilon_0=\infty$ so that $W_0^{(i)}=\frac{1}{N}$ and $\mathrm{PA}\left(\left\{W_0^{(i)}\right\},\epsilon_0\right)=1$.

- Step 0. Set $n=0$; for $i=1,\ldots,N$, sample $\theta_0^{(i)}\sim\pi(\cdot)$ and $X_{k,0}^{(i)}\sim f\left(\cdot|\,\theta_0^{(i)}\right)$ where $k=1,\ldots,M$.
- Step 1. Set $n=n+1$, if $\epsilon_{n-1}=\epsilon$ stop, otherwise determine ϵ_n by solving $\mathrm{PA}\left(\left\{W_n^{(i)}\right\},\epsilon_n\right)=\alpha\mathrm{PA}\left(\left\{W_{n-1}^{(i)}\right\},\epsilon_{n-1}\right)$ where

$$W_n^{(i)}\propto W_{n-1}^{(i)}\frac{\sum_{k=1}^{M}\mathbb{I}_{A_{\epsilon_n,y}}\left(X_{k,n-1}^{(i)}\right)}{\sum_{k=1}^{M}\mathbb{I}_{A_{\epsilon_{n-1},y}}\left(X_{k,n-1}^{(i)}\right)}. \tag{9}$$

  If $\epsilon_n<\epsilon$ then set $\epsilon_n=\epsilon$.
- Step 2. If $\mathrm{ESS}\left(\left\{W_n^{(i)}\right\}\right)<N_T$ then resample N particles from

$$\widehat{\pi}_{\epsilon_n}(d(\theta,x_{1:M})|y)=\sum_{i=1}^{N}W_n^{(i)}\delta_{\left(\theta_{n-1}^{(i)},X_{1:M,n-1}^{(i)}\right)}(d(\theta,x_{1:M})) \tag{10}$$

  also denoted abusively $\left\{\theta_{n-1}^{(i)},X_{1:M,n-1}^{(i)}\right\}$ and set $W_n^{(i)}=\frac{1}{N}$.
- Step 3. For $i=1,\ldots,N$, sample $\left(\theta_n^{(i)},X_{1:M,n}^{(i)}\right)\sim K_n\left(\left(\theta_{n-1}^{(i)},X_{1:M,n-1}^{(i)}\right),\cdot\right)$ if $W_n^{(i)}>0$ and return to Step 1.

Note that in this context, $\pi_{\epsilon_n}(\theta, x_{1:M}|y)$ can be approximated by both the weighted measures associated to $\left\{W_n^{(i)}, \left(\theta_{n-1}^{(i)}, X_{1:M,n-1}^{(i)}\right)\right\}$ as in Eq. (10) or using $\left\{W_n^{(i)}, \left(\theta_n^{(i)}, X_{1:M,n}^{(i)}\right)\right\}$. For a convergence analysis, see the discussion in [9]. This algorithm can be straightforwardly extended to the case where the distance metric only takes discrete values using the strategy presented in Section 3.1.

4 Application

The Matlab code for the toy example and the C++ code and data for the population genetics example are available at `http://www.cs.ubc.ca/~arnaud/smcabc.html`.

4.1 A Toy Example

We begin with the toy example in [4, 18]. The model is of the form

$$\theta \sim \mathcal{U}_{(-10,10)}, \;\; f(x|\theta) = 0.5\phi(x;\theta,1) + 0.5\phi(x;\theta,1/100).$$

$\mathcal{U}_{(a,b)}$ denotes the uniform distribution on the interval (a,b) and $\phi(u;m,\sigma^2)$ is the one-dimensional normal density of mean m and variance σ^2. It is assumed $y = 0$ is observed, so that the posterior density of interest is

$$\pi(\theta|x) \propto (\phi(x;0,1) + \phi(x;0,1/100)) \;\; \mathbb{I}_{(-10,10)}(\theta).$$

An ABC strategy is used to estimate $\pi(\theta|x)$, with $\eta(x) = x$ and $\rho(x,y) = |x-y| = |x|$. In this case, we have

$$\pi_\epsilon(\theta|y) \propto (\Phi(\epsilon-\theta) - \Phi(-(\epsilon+\theta)) + \Phi(10(\epsilon-\theta)) - \Phi(-10(\epsilon+\theta))) \;\; \mathbb{I}_{(-10,10)}(\theta)$$

where $\Phi(u)$ is the cumulative distribution function of the standard normal [4]. For $\epsilon = 0.025$, it is shown in [4] that $\pi(\theta|x)$ is indistinguishable from $\pi_\epsilon(\theta|y)$.

The adaptive SMC algorithm is run using a random walk MH kernel. This is based on a normal proposal of variance given, at time $n \geq 1$, by twice the empirical variance of the $\{\theta_{n-1}^{(i)}\}$ ([4]). Our experiments use $N \in \{1000, 10000, 100000\}$ particles, $M = 1$ and the adaptive SMC algorithm is run for $\alpha \in \{0.9, 0.95, 0.99\}$ and $N_T = N/2$.

In Table 1, the CPU times are given for this adaptive SMC algorithm averaged over 50 realisations using a PC Intel 3.33GHz

N / α	0.90	0.95	0.99
1000	0.3	0.5	2.3
10000	1.0	1.9	9.7
100000	10.7	22.1	112.3

Table1: Averaged CPU times in seconds for various values of N and α

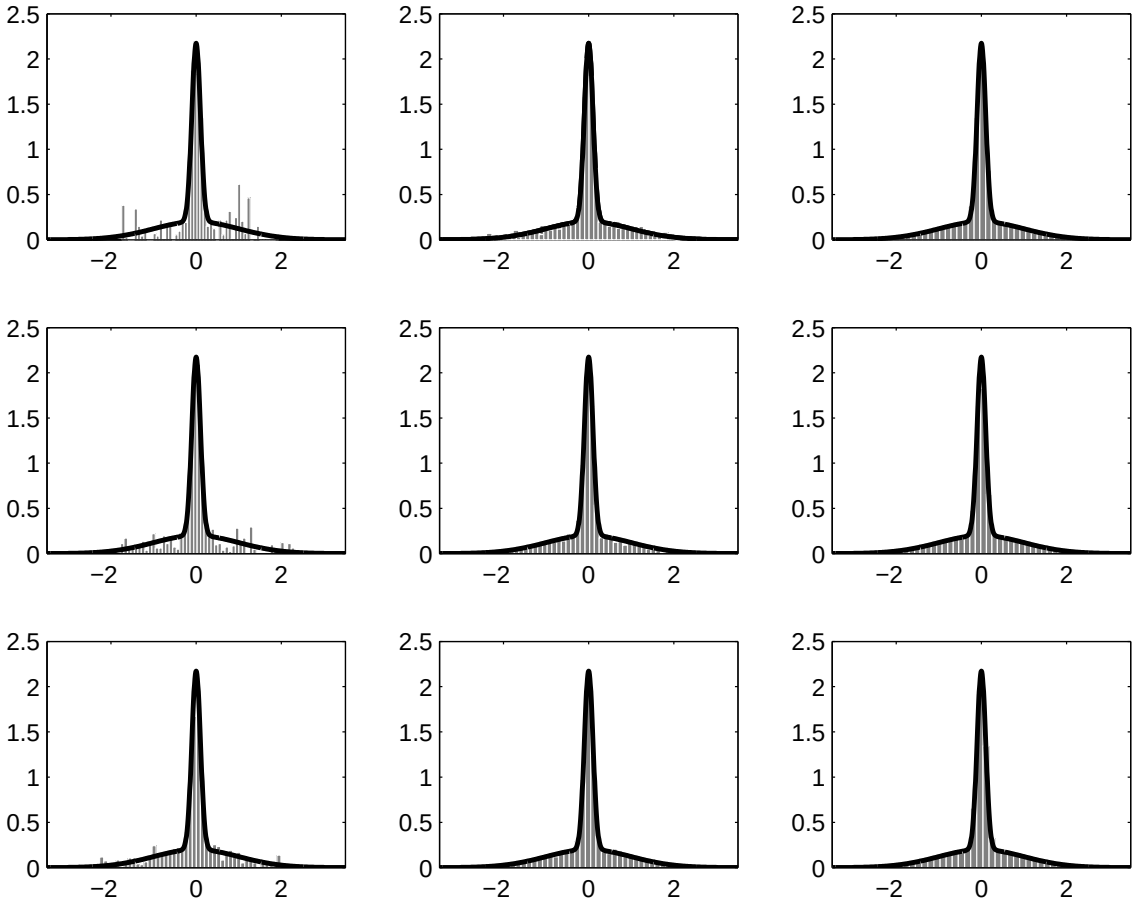
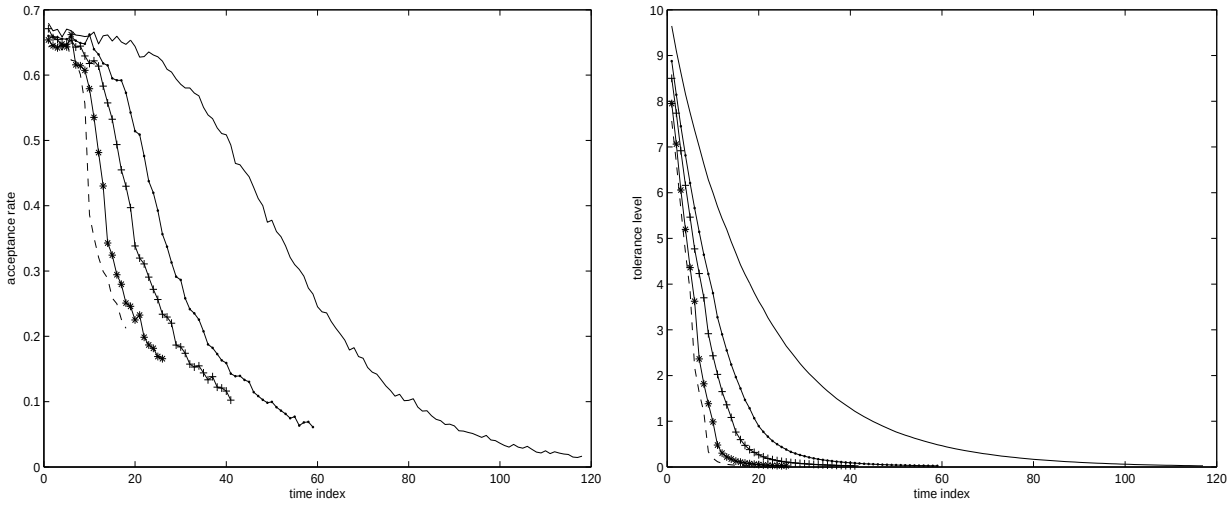


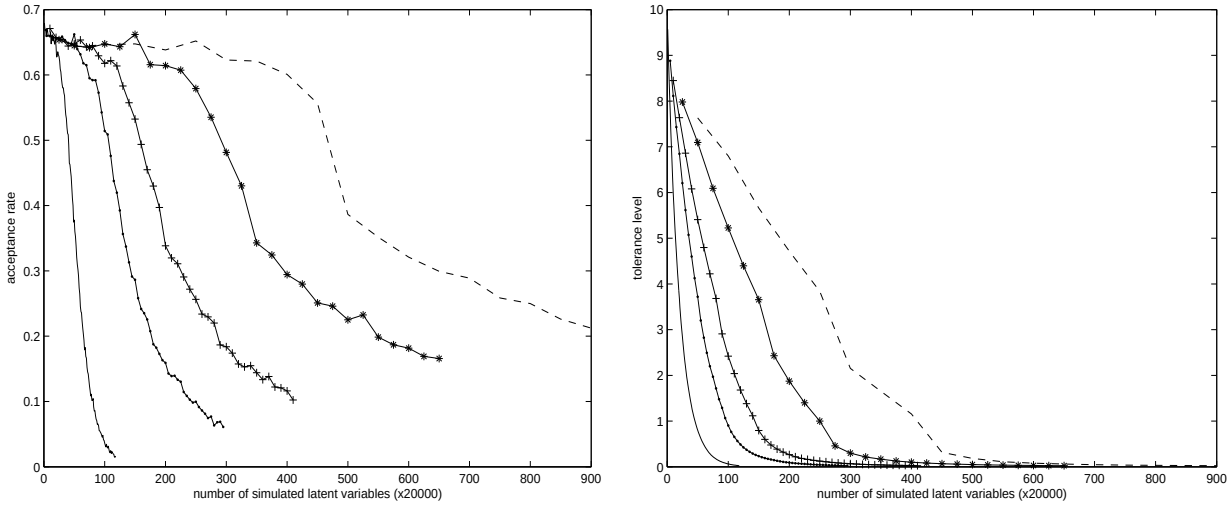
Figure 1: Histograms of the samples obtained through SMC. Each row corresponds to a different α (from top to bottom $\alpha = 0.9$, 0.95 and 0.99), each column corresponds to a different N (from left to right $N = 1000$, 10000 and 100000). The true target density $\pi_\epsilon(\theta|\, y)$ is displayed in black.

In Figure 1 we display the histograms of the samples obtained by the adaptive SMC method for these various configurations. As expected, the results improve as both α and N increases. For $N = 10000$ and $\alpha = 0.95$ we now investigate the influence of $M \in \{1, 10, 50, 100\}$ on the performance of the algorithm. In Figure 2 the average, over the alive particles, acceptance rate of the MH step and the sequence of tolerance levels $\{\epsilon_n\}$ as a function of the time index n are displayed. Note that M has a significant influence on the number of intermediate distributions required to reach the target. The higher M the smaller this number is and, as expected, the higher M the higher the average acceptance rate for a fixed ϵ. We can increase M further but the results appear very similar to $M = 50$. In general, the number M necessary to observe this stabilization depends on $f(x|\,\theta)$. The more diffuse $f(x|\,\theta)$ is (in x), the higher M should be. In Figure 3, we display again the acceptance rate and the sequence of tolerance levels as a function of the number of latent variables X simulated.



(a) Acceptance rate (b) Tolerance level

Figure 2: Average acceptance rate of the MH step (left) and sequence of tolerance levels $\{\epsilon_n\}$ (right) as a function of n for $M = 1$ (solid), 5 (dots), 10 (crosses), 25 (stars) and 50 (dashed dots).



(a) Acceptance rate (b) Tolerance level

Figure 3: Average acceptance rate of the MH step (left) and sequence of tolerance levels $\{\epsilon_n\}$ (right) as a function of the number of simulated latent variables ($\times 20000$) for $M = 1$ (solid), 5 (dots), 10 (crosses), 25 (stars) and 50 (dashed dots).

4.2 A Population Genetics Example

Our example is focussed upon the coalescent model, in the context of Bayesian inference. The example in [15] is essentially repeated. The model is composed of a stochastic tree in continuous time, which details the ancestry of a collection of individuals of a given genetic type; see [20] for details. The tree is comprised of the genetic events:

- coalescence, where individuals combine,
- mutation, where the type of an individual changes,

along with ancestry and topology of the tree. The observations are sequence data, i.e. for some $m \in \mathbb{N}$ $\mathsf{Y} = \{1, \ldots, d\}^m$, $y \in \mathsf{Y}^n$. For example, for DNA sequences, we have $d = 4$. In this case, a prior is placed on θ, the mutation rate and the posterior density of interest is

$$\pi(\theta, \mathcal{G}|y) \propto f(y|\theta, \mathcal{G})\pi(\mathcal{G}|\theta)\pi(\theta)$$

where $\mathcal{G}$ is the genealogy (topology of the tree and coalescent times). The likelihood term $f(y|\theta)$ can be calculated, assuming independence across sites, by the peeling algorithm [10]. In our model, mutations occur by picking a site, i.e. one of the $1, \ldots, d$, uniformly at random, and mutating using a $d \times d$ transition matrix P. If the length of the DNA sequence, m, and the number of data, n, is very large, the computational cost of calculating the likelihood is very high; ABC methods are useful in this context.

4.2.1 The Data

The data used is as in [15]. These are Nuu Chah Nulth mtDNA data, comprised of $n = 63$ data points of sequence lengths of 360bp. Of the 63 DNA sequences, 28 of them differ, and of the 360 sites, 26 of them differ (termed segregating sites). The base frequencies are $(\pi_A, \pi_C, \pi_T, \pi_G)$ are (0.330, 0.112, 0.337, 0.221), which are used as initial probabilities in the tree. More thorough description can be found in [22].

4.2.2 Model on an Extended State-Space

In the ABC-MCMC algorithm of [15], it is noted that the genealogy is not enough to yield reasonable acceptance rates. As a consequence, the authors include additional information on the tree, including mutation times and the sites which mutate. We do the same here.

It is assumed that the site of a mutation is uniformly distributed, given that a mutation has occurred. Given k split (coalescent) or mutation events, write the event times, as $t_{1:k}$, $b_{1:k}$ as the branch of which the event occurs, $i_{1:k}$ as the indicator of a split, $s_{1:k} \in \{0, \ldots, 360\}$

as a site (here 0 is associated to the fact that no mutation occurred), the coalescent prior is, up to proportionality

$$\left\{ \prod_{i=1}^{k} p(t_j|i_{1:j-1};\theta)p(i_j|i_{1:j-1};\theta)p(b_j|i_{1:j-1})p(s_j|i_j) \right\} \mathbb{I}_{\{n-1\}}\left(\sum_{j=1}^{k} i_j \right)$$

with

$$T_j|i_{1:j-1},\theta \sim \mathcal{E}x\left(0.5\big(\sum_{l=1}^{j-1} i_l + 2\big)\big(\sum_{l=1}^{j-1} i_l + 1 + \theta\big) \right)$$

with $\mathcal{E}x$ the exponential density and the other densities uniform on the space unless there is no mutation, on which the S_j, i_j is 0 with probability 1.

Our target density is

$$\pi_\epsilon(\theta, x_{1:M}, u_{1:M}|y) \propto \left\{ \frac{1}{M} \sum_{i=1}^{M} \mathbb{I}_{A_{\epsilon_n,y}}(x_i) \right\} \left[\prod_{i=1}^{M} f(x_i|\theta)p(u_i|\theta) \right] \pi(\theta)$$

where the genealogy and auxiliary variables are denoted u and the prior on θ is an exponential distribution with rate 1.5. The summary statistic is the number of segregating sites and the distance metric the absolute value between the number of segregating sites. We take $\epsilon = 1$.

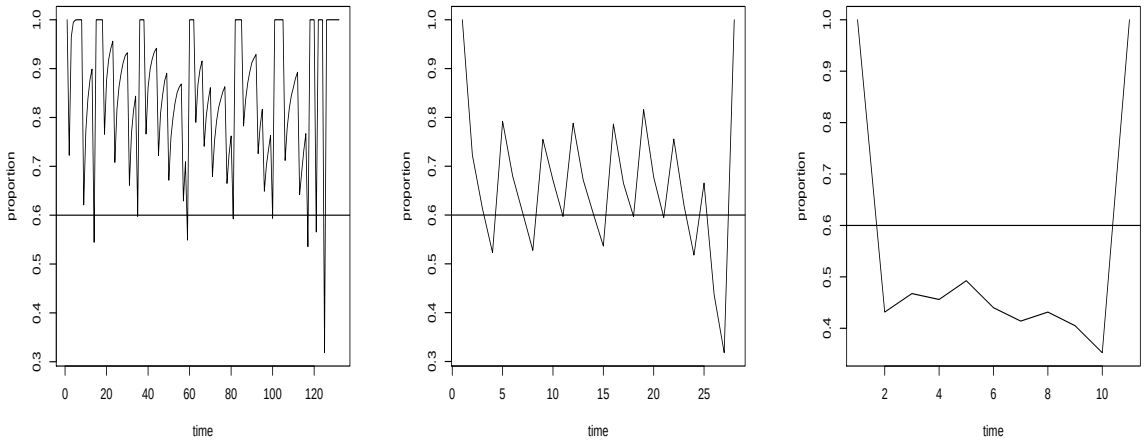
4.2.3 MCMC Kernels

In order to move the particles around the space two simple MH moves are used. The first one is to perturb θ with a log-normal random walk, and to re-simulate all of the states from the prior. The second is the birth/death of a mutation. In the birth, we propose to add a mutation, uniformly at random from the tree, with the death doing the opposite. The acceptance probability of this move is easily calculated, in the spirit of [12], and is omitted for brevity.

4.2.4 Simulation Details

Our adaptive SMC algorithm is run with $M = 3$, $P = (1/d)\mathbf{1}_{d\times d}$ and $\mathbf{1}_{d\times d}$ the $d \times d$ matrix of 1's, $\beta = 0.6$ (the resampling threshold for the PA, see Section 3.1) and $\alpha \in \{0.7, 0.8, 0.95\}$, $S = 10$ and $\tau \in \{0.75, 0.75, 0.9\}$, $\tau' \in \{0.1, 0.1, 0.1\}$. The value of ϵ_n^* is decreased by 1 when attempting to find the new value of ϵ_n. The stability of the results w.r.t $N \in \{2000, 5000, 10000\}$ is investigated.

In our simulations, it was found for small M ($M < 3$) that the results were worse; the PA falls much more quickly. The results do improve with M, but we would suggest that for very large M, the algorithm is too slow for this example to compensate the improvement in performance.



(a) $\alpha = 0.95$ (b) $\alpha = 0.8$ (c) $\alpha = 0.7$

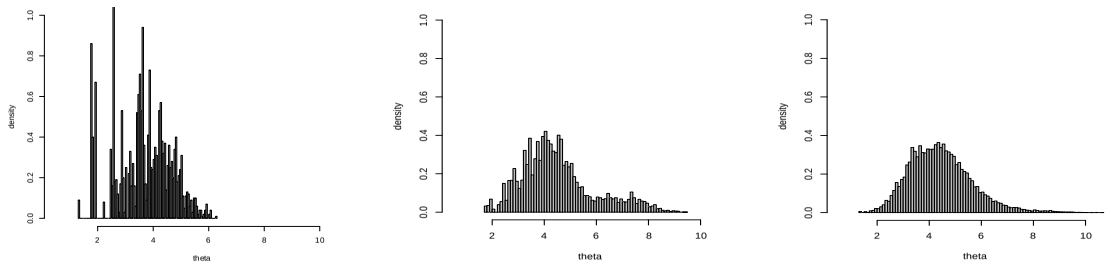
Figure 4: Results from the ABC Simulations. These are the proportion of samples alive for the 3 settings of α at each time-step of the algorithm. The horizontal line is the resampling threshold.

One criticism is in terms of setting the parameters α, τ, τ', S. It should be noted, however, that the implications of the choice of these parameters are well-understood (Section 3.1) and it is far easier than selecting the $\{\epsilon_n\}$ in the first place.

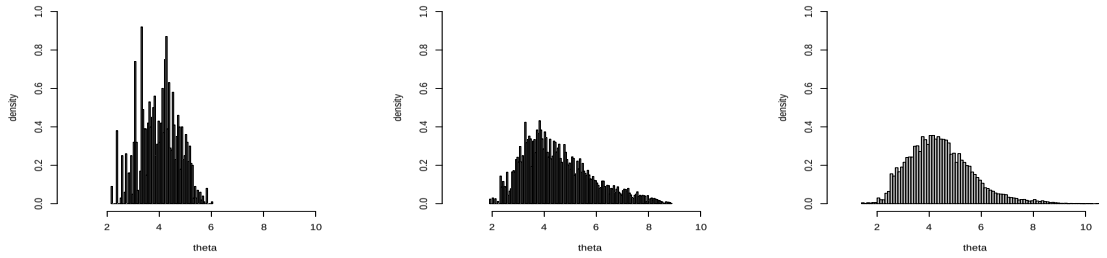
4.2.5 Simulation Results

The results can be seen in Figures 4-6. Unless otherwise stated, the results are for $N = 2000$. In Figure 4 the proportion of surviving particles is plotted against the time parameter, for each setting of α. This is recorded before a resampling step may occur - i.e. the particles will be sampled afterwards. It is immediately observed that $\alpha \in \{0.7, 0.8\}$ is run for fewer time-steps; the CPU times for $\alpha \in \{0.7, 0.8, 0.95\}$ are 232, 315 and 673 seconds, respectively. On the basis of this plot, it is quite clear, for this example that α should be large: 673 seconds is not a long run time and the weights degenerate much less than for the other two settings. For $\alpha = 0.95$, the performance is quite acceptable, with a slow decline, but also a move upwards, when the algorithm does not decrease ϵ_n (see Figure 6 (a)). In this case, the algorithm uses the kernels to bring 'dead particles' back to life, in the hope that the drop in PA is not substantial. This is not completely successful, as we can also see a slight drop in surviving particles. Whilst this is not ideal, the algorithm allows a sizeable number of particles to explore those densities.

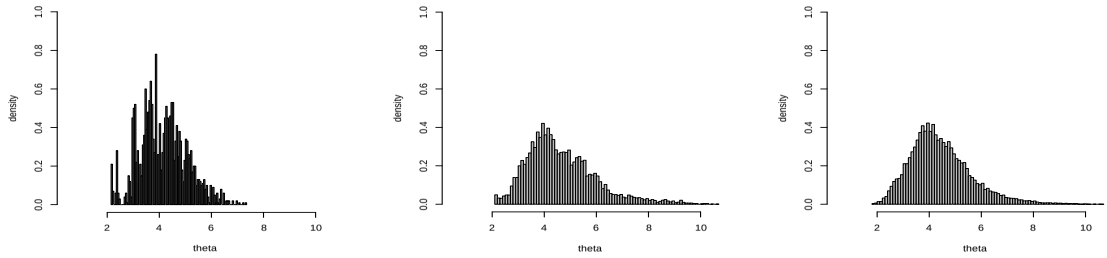
In Figure 5, the histogram of the θ from the posterior are given, with $N \in \{2000, 5000, 10000\}$. It should be noted that these samples were resampled, and thus, 5 extra MCMC steps for each particle were allowed. The conclusions of the preceding paragraph appear to be confirmed,



(a) $\alpha = 0.7$

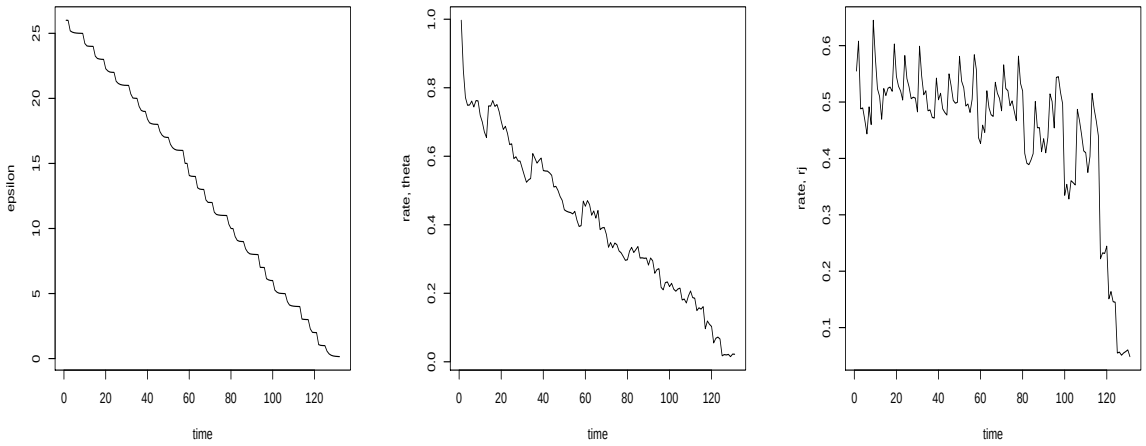


(b) $\alpha = 0.8$



(c) $\alpha = 0.95$

Figure 5: Histograms of the samples obtained through SMC. Each row corresponds to a different α (from top to bottom $\alpha = 0.70, 0.80, 0.95$), each column corresponds to different N (from left to right $N = 2000$, 5000, 10000).



(a) ϵ (b) Random Walk (c) Reversible Jump

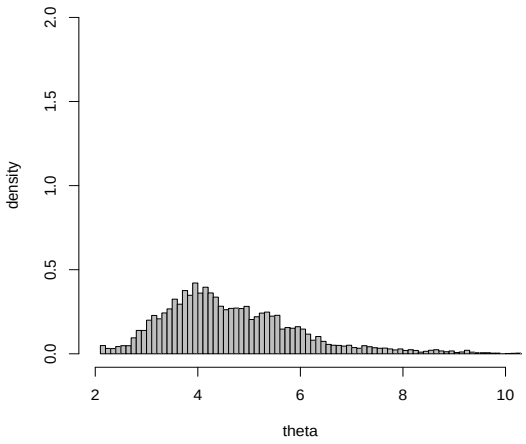
Figure 6: Results from the ABC Simulations. The plots are the acceptance rates and evolution of ϵ_n when $\alpha = 1$.

with the samples for $\alpha = 0.95$ exhibiting far more stability w.r.t. N.

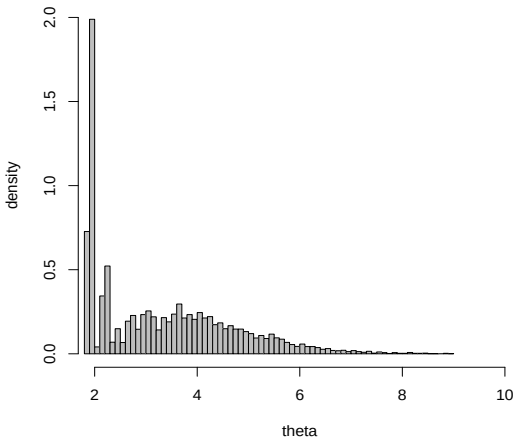
Given the results above, let us investigate the case $\alpha = 0.95$, in terms of the tolerance levels and the acceptance rates, of the MCMC steps, across the algorithm. The tolerance level in Figure 6 (a) starts at a rather high adaptively set value, 26, which is the number of segregating sites and, decreases linearly. This linear rate falls once $\epsilon = 2$, which indicates that many particles have the values of ρ around this value. The average acceptance rate of the MCMC moves for the prior and reversible jump proposals are displayed in Figures 6 (b)-(c).

The faster the MCMC kernel mixes, the better the results of the SMC algorithm will be. However, even in the event (of say) a 1% acceptance rate of the MCMC algorithm for ϵ, our SMC algorithm substantially improves over the MCMC results. This can be seen in Figure 7. In this case, an MCMC algorithm with the same MCMC settings and CPU time (3141 seconds) at the SMC algorithm using $\alpha = 0.95$ and $N = 10000$ was run for 590000 steps of the two moves, and every 59^{th} sample was recorded to obtain 10000 samples. The representation of the posterior on θ is much worse for the MCMC algorithm, and we were unable to improve it, by changing proposal variances. The MCMC gets trapped around values of $\theta = 2$ for which very few simulated summary statistics are compatible with the data.

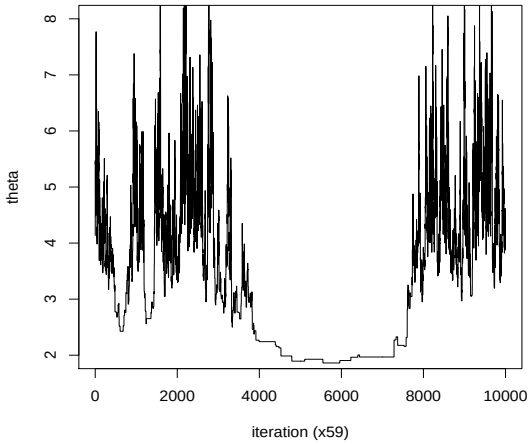
For comparison, with the inferential results of [15], the expectation of the most recent common ancestor, computed using $\alpha = 0.95$ and $N = 10000$, is 2.44. This is quite similar to the values obtained in that paper (1.82), for this data, but for a different prior parameter on θ.



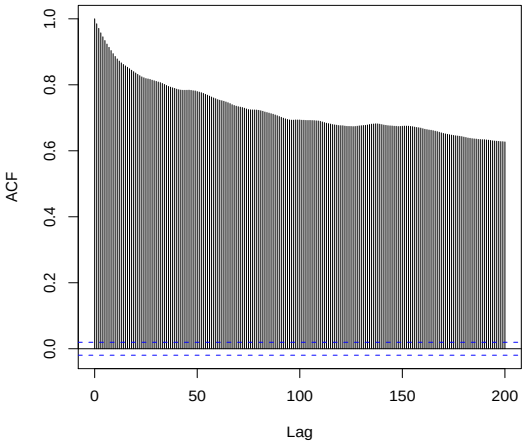
(a) SMC approximation of the target



(b) MCMC approximation of the target



(c) Trace plot



(d) Autocorrelation function

Figure 7: SMC (top left) versus MCMC (top right) approximation of the target for the same CPU time. Trace plot of the MCMC run (bottom left) and associated autocorrelation function (bottom right).

5 Discussion

In this paper we have presented an adaptive SMC algorithm for ABC. Our approach has a computational cost that is linear in the number of samples and is able to adaptively calculate the tolerance levels in a sensible manner. It appears possible to produce accurate answers with little user input. However, for a fixed computational complexity, it can be difficult to *a priori* decide what the best combination of parameters N, M and α is; it is highly model dependent.

We have not provided any convergence results in this paper. Precise convergence results for adaptive SMC methods have been recently obtained in [9]. The results there and in [7, Theorem 7.4.3.] can be combined to give rates of convergence for our algorithm.

Acknowledgement

We would like to thank Dave Stephens and Maria De Iorio for some useful discussion related to the example.

References

[1] Andrieu, C. & Johanes, T. (2008). A tutorial on adaptive Markov chain Monte Carlo methods. *Statist. Comput.*, to appear.

[2] Andrieu, C., Berthelesen, K., Doucet, A. & Roberts, G.O. (2008). The expected auxiliary variable method for Monte Carlo simulation. In preparation.

[3] Beaumont, M. A., Zhang, W. & Balding, D. (2002). Approximate Bayesian computation in population genetics. *Genetics*, **162**, 2025–2035.

[4] Beaumont, M. A., Cornuet, J. M., Marin J. M., & Robert, C. P. (2008). Adaptivity for ABC algorithms: the ABC-PMC scheme. Technical report, Université Paris Dauphine.

[5] Chopin, N. (2002). A sequential particle filter for static models. *Biometrika*, **89**, 539–552.

[6] Crooks, G.E. (1998) Nonequilibrium measurements of free energy differences for microscopically reversible Markovian systems. *J. Stat. Phys.*, **90**, 1481-1487.

[7] Del Moral, P., (2004). *Feynman-Kac formulae. Genealogical and interacting particle systems*, Springer-Verlag: New York.

[8] Del Moral, P., Doucet, A. & Jasra, A. (2006). Sequential Monte Carlo samplers. *J. Roy. Statist. Soc.* B, **68**, 411–436.

[9] Del Moral, P., Doucet, A. & Jasra, A. (2008). On adaptive resampling strategies for sequential Monte Carlo methods. Technical report 00332436, INRIA.

[10] Felesenstein, J. (1981). Evolutionary trees from DNA sequences: A maximum likelihood approach. *J. Mol. Evol.*, **17**, 368-376.

[11] Gilks, W.R. and Berzuini, C. (2001). Following a moving target - Monte Carlo inference for dynamic Bayesian models. *J. Roy. Statist. Soc.* B, **63**, 127–146.

[12] Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, **82**, 711–732.

[13] Kitagawa, G. (1996) Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *J. Comp. Graph. Statist.*, **5**, 1-25.

[14] Liu, J.S. (2001) *Monte Carlo Strategies in Scientific Computing*. New York: Springer Verlag.

[15] Majoram, P., Molitor, J., Plagnol, V. & Tavaré. S. (2003). Markov chain Monte Carlo without likelihoods. *Proc. Natl. Acad. Sci.*, **100**, 15324–15328.

[16] Neal, R. M. (2001). Annealed importance sampling. *Statist. Comp.*, **11**, 125–139.

[17] Peters, G. W., Fan, Y. & Sisson, S. A. (2008). On sequential Monte Carlo, partial rejection control and approximate Bayesian computation. Technical report, University of South Wales.

[18] Sisson, S., Fan, Y. & Tanaka, M. M. (2007). Sequential Monte Carlo without likelihoods. *Proc. Natl. Acad. Sci.*, **104**, 1760–1765.

[19] Sisson, S., Fan, Y. & Tanaka, M. M. (2008). A note on backward kernel choice for sequential Monte Carlo without likelihoods. Technical report, University of New South Wales.

[20] Stephens, M. & Donelly, P. (2000). Inference in molecular population genetics (with discussion). *J. Roy. Statist. Soc.* B, **62**, 605–655.

[21] Toni, T., Welch, D., Strelkowa, N., Ipsen, A. & Stumpf, M.P.H. (2008) Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *J. Roy. Soc. Interface*, (in press).

[22] Ward, R. H., Frazier, B. L., Dew-Jagner, K. & Paabo, S. (1991). Extensive mitochondrial diversity within a single Amerindian tribe. *Proc. Natl. Acad. Sci.*, **88**, 8720-8724.