Adaptive Independence Samplers

J. M. Keith [a,1,2], D. P. Kroese [b,1], G. Yu. Sofronov [b,1,*]

[a] *School of Mathematical Sciences, Queensland University of Technology, GPO Box 2434 Brisbane, Qld 4001, Australia*

[b] *Department of Mathematics, The University of Queensland, Brisbane, Qld 4072, Australia*

Abstract

Markov chain Monte Carlo (MCMC) is an important computational technique for generating samples from non-standard probability distributions. A major challenge in the design of practical MCMC samplers is to achieve efficient convergence and mixing properties. One way to accelerate convergence and mixing is to adapt the proposal distribution in light of previously sampled points, thus increasing the probability of acceptance. In this paper, we propose two new adaptive MCMC algorithms based on the Independent Metropolis-Hastings algorithm. In the first, we adjust the proposal to minimize an estimate of the cross-entropy between the target and proposal distributions, using the experience of pre-runs. This approach provides a general technique for deriving natural adaptive formulae. The second approach uses multiple parallel chains, and involves updating chains individually, then updating a proposal density by fitting a Bayesian model to the population. An important feature of this approach is that adapting the proposal does not change the limiting distributions of the chains. Consequently, the adaptive phase of the sampler can be continued indefinitely. We include results of numerical experiments indicating that the new algorithms compete well with traditional Metropolis-Hastings algorithms. We also demonstrate the method for a realistic problem arising in Comparative Genomics.

Key words: Markov chain Monte Carlo, generalized Markov sampler, adaptive methods, cross-entropy, comparative genomics

* Corresponding author.
Email addresses: j.keith@qut.edu.au (J. M. Keith), kroese@maths.uq.edu.au (D. P. Kroese), georges@maths.uq.edu.au (G. Yu. Sofronov).

[1] Supported by the Australian Research Council (grant number DP0556631)
[2] Supported by the NHMRC grant 'Statistical methods and algorithms for analysis of high-throughput genetics and genomics platforms' (389892)

1 Introduction

Markov chain Monte Carlo is an enabling computational technique that provides a means of generating samples from an arbitrary statistical distribution. The need for such sampling is common in Bayesian analyses, usually for the purpose of estimating the value of an integral containing a posterior distribution. However, the need to sample from a distribution also arises in non-Bayesian contexts.

The first MCMC sampler was invented by Metropolis *et al.* [22]. It was subsequently generalized by Hastings and this important generalization is known as the Metropolis-Hastings algorithm [16]. The sampler begins with an arbitrary value x_0 and generates a Markov chain in the following manner. Using an arbitrary transition function $Q(x, y)$ (which takes the form of a density in a continuous space or a transition matrix in a discrete space), the algorithm alternates between a) proposing a new element drawn from the *proposal distribution* $Q(x, \cdot)$, where x is the *current element*, and b) either accepting the new element or repeating the current element, with the probability of acceptance given by the *acceptance ratio* $\alpha(x, y)$. Under weak conditions on Q, the chain converges asymptotically to the *target distribution* f that one wishes to sample. When a sampler is implemented on a computer with finite precision, convergence to within that precision occurs in a finite time period. This period is known as *burn-in*.

The art of designing an efficient Metropolis-Hastings algorithm lies chiefly in choosing an appropriate proposal distribution. The length of the burn-in period, and the speed at which the chain mixes after burn-in, depend critically on the proposal. Convergence and mixing will be slow if the proposed transitions are mostly between nearby states in state space. However, a preponderance of distant transitions may also slow the chain if it results in a lower acceptance ratio, as is likely if distant transitions are proposed indiscriminantly. Thus the proposal should ideally be chosen in such a way as to allow both distant transitions and a high acceptance ratio. One way to achieve this is to adapt the proposal distribution in light of sampled elements.

There is an extensive literature on adaptive MCMC algorithms. One approach to adapting the proposal distribution is to use pre-runs and tune the proposal based on the experience of the pre-runs, as suggested in [8,13]. Another approach uses multiple chains (e.g., [11,4,28]) and updates the proposal for iteration $t+1$ based on the current 'population' (that is, the set of current elements across all chains) at iteration t. A number of interesting ideas regarding this latter approach have recently been introduced. Warners [29] has used a kernel estimation of the target distribution to design the proposal distribution. The method in [4] is based on running an increasing number of chains. Tierney

and Mira [28] have used a delayed rejection approach.

Some adaptive MCMC algorithms use either the whole or the recent history of the chain and only a single chain for adaptation. These are either based on regeneration or slowing down the adaptation as sampling proceeds (see [3,7,12,14,15,17,23,24,26]).

A regeneration idea is proposed by Mykland, Tierney and Yu [23]. Gilks, Roberts and Sahu [12] have shown that the transition kernel used in an MCMC algorithm can be updated without damaging the ergodicity at regeneration times. The self-regenerative algorithm by Sahu and Zhigljavsky [26] is based on constructing an auxiliary chain and picking elements of this chain a suitable number of times.

Haario, Saksman and Tamminen [14] have proposed an adaptive version of the random walk Metropolis algorithm where the covariance matrix of the proposal kernel is sequentially updated. To solve high-dimensional problems, Haario, Saksman and Tamminen [15] have introduced an adaptive MCMC algorithm in which the adaptation is performed componentwise. The algorithm uses Gaussian proposal distributions whose variances depend on time. Atchade and Rosenthal [3] have considered adaptive MCMC algorithms that generate stochastic processes based on sequences of transition kernels, where each transition kernel is allowed to depend on the history of the process. Gasemyr [7] has presented an adaptive, non-Markovian version of the Metropolis-Hastings algorithm with a fixed parametric class of proposal distributions, where the parameters of the proposal distribution are updated periodically on the basis of the recent history of the chain. Holden [17] has proved convergence of regenerative chains using part of the history and adaptive chains using the full history. Pasarica and Gelman [24] have developed an adaptive algorithm that uses the information accumulated by a single path and adapts the choice of the parametric kernel in the direction of the local maximum of the objective function (the expected squared jumped distance) using multiple importance sampling techniques.

We consider the Independent Metropolis-Hastings Algorithm to sample from a target density function $f(x)$. This algorithm can be written as follows:

Given the current state x_t*:*

(1) Generate $Y \sim g(y)$.
(2) Generate $U \sim \mathsf{U}(0,1)$ *and deliver*

$$x_{t+1} = \begin{cases} Y, & \text{if } U \leqslant \alpha(x_t, Y) \\ x_t, & \text{otherwise,} \end{cases}$$

where

$$\alpha(x, y) = \min\{\varrho(x, y), 1\},$$

with

$$\varrho(x, y) = \frac{f(y)g(x)}{f(x)g(y)}.$$

When the proposal g is equal to f, the acceptance ratio is one and the correlation between adjacent elements of the chain is zero. Thus it is desirable to choose the proposal distribution g as closely as possible to the target f. One possibility is to sample from conditional distributions of f, a technique known as *Gibbs sampling*. However, this technique can be computationally inefficient for some problems. In an adaptive framework, a natural strategy is to adjust a trial proposal g_0 to obtain a new proposal g that is 'closer' to f.

In this paper, we propose two new adaptive MCMC algorithms, which we call *Adaptive Independence Samplers (AIS)*. The first approach, which can be considered to belong to the category of MCMC algorithms that use pre-runs, is as follows. We generate a sample from a target density via some proposal density using the Independent Metropolis-Hastings Algorithm and then use these data to update the proposal in a similar manner to that used in the cross-entropy (CE) method [25]. We call this method the *Cross-Entropy Adaptive Independence Sampler (CEAIS)*. We suggest the choice of a mixture density as a proposal. Using CEAIS, convenient update formulae for the parameters of the mixture density are obtained. The significance of using a mixture as a proposal is that a multi-modal target distribution can be approximated more closely, thus increasing the acceptance ratio. As our results demonstrate, this is an efficient tool to fit the proposal to the target density. A new idea is to choose a proposal density such that the Kullback-Leibler distance between the proposal density and the target density is minimal. While the use of pre-runs and mixture proposals is not new, the use of cross-entropy in this context is novel.

The second, Bayesian, approach is based on multiple chains, in which we update observations one by one. We periodically update a proposal density using the set of current observations across all chains. We refer to this method as the *Bayesian Adaptive Independence Sampler (BAIS)*. Again the idea of using parallel chains to obtain information for the purpose of adapting the proposal is not new, but our Bayesian approach is novel. The technique of constructing this algorithm is based on the methodology of the Generalized Markov Sampler [18]. We prove that the target distribution is the limiting distribution of the corresponding process. A critical point in constructing adaptive MCMC algorithms is that adaptation may break the Markovian property so that ergodicity cannot be guaranteed, or may change the limiting distribution of the process. However, an important property of the Bayesian method described here is that it retains the Markovian property and the target distribution as

its limiting distribution.

The paper is structured as follows. In Sections 2 and 3, we develop the two new algorithms CEAIS and BAIS, respectively. Section 2 includes a description of a variant of CEAIS that is based on the EM algorithm. In Section 4, we explain two measures of the efficiency of an MCMC algorithm that we use in Section 5 to compare CEAIS and BAIS to each other and to two non-adaptive Metropolis-Hastings samplers. Section 5 presents the results of three numerical experiments used to evaluate the new samplers. This section also includes an application of BAIS to a problem that arises in Comparative Genomics - specifically, fitting a mixture of beta distributions to a data set obtained by a sliding window analysis of conservation. Our conclusions are in Section 6.

2 CEAIS

One possible way to find an efficient proposal is to generate a sample $X_1, \ldots, X_N$ from the target f via some trial proposal g_0 (using the Independence Sampler) and then fit a new distribution g to these data. It may seem superfluous to update g_0 to g in this manner, given that $X_1, \ldots, X_N$ have already been generated. However, there are a number of reasons why it is advantageous to do so. Firstly, if a large sample is required, it may be possible to generate this sample much more efficiently by first using a small pre-run to obtain a proposal g close to the target f. Secondly, and more importantly, it is not necessary to wait for burn-in to obtain the sample $X_1, \ldots, X_N$. Even if this sample is not strictly drawn from f (and it will not be if it is obtained during burn-in) it may nevertheless be possible to obtain a new proposal g closer to f. Moreover, this process can be iterated, with the possibility of accelerated convergence. Convergence of the chain to the target distribution f is still assured provided that one stops updating the proposal after some number of iterations. Although the number of times the proposal should be updated, the number of samples to use for each update, and whether and how to subsample for the purpose of fitting the new proposal are important implementation details, we focus here on a new method of generating update formulae for the proposal.

Let g_0, g be proposal densities, and f be the target density. In practice, f can be a rather complicated function with multiple modes. We shall therefore assume that g_0 and g are mixture densities:

$$g(x) = g(x \mid \theta) = \sum_{c=1}^{k} w_c \, h_c(x, \eta_c), \quad \sum_{c=1}^{k} w_c = 1, \quad w_c \geqslant 0,$$

where $\theta = (w, \eta)$ is a parameter vector, which includes the weights $w =$

$(w_1, \ldots, w_k)$ and the vector $\eta = (\eta_1, \ldots, \eta_k)$ containing all the parameters of the densities $h_c(x, \eta_c)$, $c = 1, \ldots, k$.

We propose choosing a density g such that the distance between g and the target density f is minimal. A particular convenient distance between two densities is the Kullback-Leibler Cross-entropy distance [19], or more concisely cross-entropy (CE). The distance is defined as:

$$\mathcal{D}(f, g) = \int f(x) \ln \left(\frac{f(x)}{g(x)} \right) dx.$$

Minimizing the distance is, in our case, equivalent to solving the maximization problem

$$\max_{\theta} \int f(x) \ln \sum_{c=1}^{k} w_c \, h_c(x, \eta_c) \, dx.$$

As in the CE method [25], we can estimate the optimal solution θ^* as the solution of the program

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} f(X_n) \ln \sum_{c=1}^{k} w_c \, h_c(X_n, \eta_c),$$

where a sample $X_1, \ldots, X_N$ is generated from f via g_0 (using the Independence Sampler).

The variable X_n can be obtained as a component of the pair (X_n, C_n) by first choosing C_n with probability w_{C_n} and then generating an observation from h_{C_n}. We are interested in inference for X_n, so C_n may be considered a nuisance parameter [9]. Using Lagrange multipliers and the fact that $\sum_c w_c = 1$, the solution to the constrained maximisation problem is:

$$\widehat{w}_c = \frac{1}{N} \sum_{n=1}^{N} I_{\{C_n = c\}}.$$

For the Gaussian case, this means that h_c is a normal density with parameters (μ_c, Σ_c), we have

$$\widehat{\mu}_c = \frac{\sum_{n=1}^{N} I_{\{C_n = c\}} X_n}{\sum_{n=1}^{N} I_{\{C_n = c\}}},$$

and

$$\widehat{\Sigma}_c = \frac{\sum_{n=1}^{N} I_{\{C_n = c\}} (X_n - \widehat{\mu}_c)(X_n - \widehat{\mu}_c)^T}{\sum_{n=1}^{N} I_{\{C_n = c\}}}.$$

An alternative approach is to estimate θ using the well-known EM method (see, e.g., [5,21]). It can be shown that

$$\widehat{w}_c = \frac{1}{N} \sum_{n=1}^{N} q^o(c \,|\, X_n),$$

where

$$q^o(c \mid X_n) = \frac{w_c\, h(X_n, \eta_c^o)}{g_0(X_n)}, \quad c = 1, \ldots, k.$$

For the Gaussian case this leads to the formulas

$$\widehat{\mu}_c = \frac{\sum_{n=1}^N q^o(c \mid X_n) X_n}{\sum_{n=1}^N q^o(c \mid X_n)},$$

and

$$\widehat{\Sigma}_c = \frac{\sum_{n=1}^N q^o(c \mid X_n)(X_n - \widehat{\mu}_c)(X_n - \widehat{\mu}_c)^T}{\sum_{n=1}^N q^o(c \mid X_n)}.$$

When these latter updating formulae are used, we refer to the resulting sampler as the *Expectation-Maximization Adaptive Independence Sampler (EMAIS).*

To choose the number of mixture components k we can use information criteria such as Akaike's information criterion (AIC) [1], the Bayesian information criterion (BIC) [27], or the Deviance information criterion (DIC) [9]. These are popular scoring functions used to rank competing models on the basis of a compromise between goodness of fit and model complexity. The simplest of these is the AIC, which can be written for a mixture model with k components as follows:

$$\begin{aligned} \mathrm{AIC}_k &:= -2\ln(\text{maximum likelihood}) + 2(\text{number of parameters}) \\ &= -2\ln\left(\prod_{n=1}^N g(X_n, \theta^{**})\right) + 2\left(k + \sum_{c=1}^k p_c\right), \end{aligned}$$

where p_c is the number of parameters of the density h_c and θ^{**} is the value of the parameter that maximizes the likelihood of $X_1, \ldots, X_N$, considered as data. As an approximation to θ^{**}, one may use the value θ^* obtained via either the Cross-Entropy or the EM method. One prefers the k-mixture model that has the lowest value of the AIC.

3 BAIS

In this section we describe the second of our adaptive independence samplers, which involves running multiple parallel chains with a common proposal distribution. The proposal is updated by fitting a Bayesian model to the population of current elements across all chains. Updating the proposal is shown to have no effect on the limiting distribution of each chain, and thus can be continued indefinitely, unlike the update procedure used in CEAIS or EMAIS.

Let f be the target pdf and let $g(x|\theta)$ be a proposal distribution, defined up to a parameter θ, which is to be updated. Let $\theta_0, \theta_1, ...$ be the parameters for the

sequence of proposals. Suppose we have N parallel chains $\{X_{1,j}, j = 1, 2, \dots\}$, $\dots$, $\{X_{N,j}, j = 1, 2, \dots\}$, which we refer to as the *sampling chains*. Using the set of current elements $(X_{1,j}, \dots, X_{N,j})$, we update θ_j at each step of the algorithm. That is, after updating each of the N chains, we update the proposal itself. Thus in effect we cycle through updates for $N+1$ chains, since $\theta_0, \theta_1, \dots$ may also be regarded as values of an underlying Markov chain, which we refer to as the *parameter chain*.

In order to describe the algorithm more precisely and to prove that the limiting distribution of each of the N sampling chains is the target distribution, we shall use the framework of the Generalized Markov Sampler [18]. Briefly, the Generalized Markov Sampler involves iterating two steps, known as the Q-step and the R-step. In the Q-step, a 'move type' is selected in accordance with a transition kernel $Q((m, z), (m', z))$ defined on a space $\mathcal{M} \times \mathcal{Z}$ obtained by augmenting a set of move types $\mathcal{M}$ to the target space $\mathcal{Z}$. In the R-step, a transition of type m' is performed in accordance with a transition function $R((m', z), (m'', z'))$, the general formulation of which is described in [18]. In what follows, we particularize the description of the Generalized Markov Sampler for the problem at hand.

Let $\mathcal{X}$ denote the target space, that is, the space on which the target distribution f is defined. Let Θ denote the space of parameters for the proposal distribution. We may regard the $N + 1$ parallel chains as a single chain defined on a space $\mathcal{Z} := \Theta \times \mathcal{X}^N$. As far as the Generalized Markov Sampler is concerned, the 'target space' is $\mathcal{Z}$. However, this ambiguity should cause no confusion, as now that we have introduced this notation we shall no longer use the term 'target space'.

The Q-step is used to propose either a new element $y \in \mathcal{X}$ or a new parameter $\theta \in \Theta$. The R-step is used to accept or reject it in accordance with an acceptance probability. More formally, the Q-step describes a transition $((i, u), z) \to ((i', u'), z)$, where $i, i' \in \{0, 1, 2, \dots, N\}$, $u, u' \in \mathcal{X} \cup \Theta$, $i' = i + 1$ or $i' = 0$, $i = N$, $z = (\theta, x_1, x_2, \dots, x_N)$,

$$Q\Big(((i, u), z), ((i', u'), z)\Big) = \begin{cases} g(u' \mid \theta), & u' \in \mathcal{X}, i' = i + 1, \\ & i \in \{0, 1, \dots, N - 1\}, \\ h(u' \mid x_1, x_2, \dots, x_N), & u' \in \Theta, i' = 0, i = N, \\ 0, & \text{otherwise.} \end{cases}$$

The distribution $h(u' \mid x_1, x_2, \dots, x_N)$ is the posterior probability of the parameter vector $u' \in \Theta$, treating $x_1, x_2, \dots, x_N$ as data. We consider the special case $\mathcal{X} = \Re^d$ for some positive integer d. Then we may use a multivariate

normal distribution as our proposal:

$$g(x \,|\, \theta) = \mathsf{N}(x \,|\, \mu, \Sigma) \propto |\Sigma|^{-1/2} \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\}.$$

Using a non-informative prior, as in [9], we obtain the posterior distribution:

$$\begin{aligned} &h(\theta \,|\, x_1, \ldots, x_N) = h(\mu, \Sigma \,|\, x_1, \ldots, x_N) = \mathsf{N}(\mu \,|\, \bar{x}, \Sigma/N) \cdot \mathsf{Inv\text{-}W}_{N-1}(\Sigma \,|\, S), \\ &S = \sum_{n=1}^{N} (x_n - \bar{x})(x_n - \bar{x})^T, \quad \bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n. \end{aligned}$$

To obtain parameters μ and Σ, we first draw Σ from an Inverse-Wishart distribution $\mathsf{Inv\text{-}W}_{N-1}(\Sigma \,|\, S)$, then we draw μ from a Normal distribution $\mathsf{N}(\mu \,|\, \bar{x}, \Sigma/N)$. For further details, see [9].

The R-step of the method depends on the stationary distribution of the transition function Q. We claim that this is:

$$q((i', u'), z) = \begin{cases} \dfrac{1}{N+1}\, g(u' \,|\, \theta), & u' \in \mathcal{X}, i' \in \{1, \ldots, N\}, \\ \dfrac{1}{N+1}\, h(u' \,|\, x_1, x_2, \ldots, x_N), & u' \in \Theta, i' = 0. \end{cases}$$

The proof follows.

Theorem 1 *The distribution $q((i', u'), z)$ is stationary with respect to Q.*

Proof. It is easily shown that

$$\sum_{i=0}^{N} Q\Big(((i,u),z),((i',u'),z)\Big) = (N+1) q((i',u'),z),\ i' \in \{0,1,\ldots,N\},\ u' \in \mathcal{X} \cup \Theta,$$

since all but one term in the summation is zero. Also, note that $Q\Big(((i,u),z),((i',u'),z)\Big)$ does not depend on u. Then

$$
\begin{aligned}
&\int q((i,u),z)Q\Big(((i,u),z),((i',u'),z)\Big)\,d(i,u)\\
&=\int \frac{1}{N+1}\,h(u\,|\,z)Q\Big(((0,u),z),((i',u'),z)\Big)\,du\\
&\quad+\sum_{i=1}^{N}\int \frac{1}{N+1}\,g(u\,|\,z)Q\Big(((i,u),z),((i',u'),z)\Big)\,du\\
&=\frac{1}{N+1}Q\Big(((0,u),z),((i',u'),z)\Big)\int h(u\,|\,z)\,du\\
&\quad+\frac{1}{N+1}\sum_{i=1}^{N}Q\Big(((i,u),z),((i',u'),z)\Big)\int g(u\,|\,z)\,du\\
&=\frac{1}{N+1}\sum_{i=0}^{N}Q\Big(((i,u),z),((i',u'),z)\Big)\\
&=q((i',u'),z).
\end{aligned}
$$

∎

The R-step involves a transition function $((i',u'),z)\to((i',u''),z')$ such that all coordinates of z' are the same as those of z except for possibly the i'-th coordinate (remembering that co-ordinate numbering starts from zero). Recalling that $z=(\theta,x_1,\ldots,x_N)$, this transition function takes the form:

$$
R\Big(((i',u'),z),((i',u''),z')\Big)=\begin{cases}
\alpha_{i'}(x_i,u'), & z'=(\theta,x_1,\ldots,u',\ldots,x_N),\\
& u'\in\mathcal{X},i'\in\{1,\ldots,N\},\\
& u''=x_i,\\
1-\alpha_{i'}(x_i,u'), & z'=z,u'\in\mathcal{X},\\
& i'\in\{1,\ldots,N\},u''=u',\\
1, & z'=(u',x_1,\ldots,x_N),\\
& u'\in\Theta,i'=0,\\
0, & \text{otherwise,}
\end{cases}
$$

where

$$
\begin{aligned}
\alpha_{i'}(x_i,u')&=\min\left\{\varrho_{i'}(x_{i'},u'),1\right\},\\
\varrho_{i'}(x_{i'},u')&=\frac{f(u')h(\theta\,|\,x_1,\ldots,u',\ldots,x_N)g(x_{i'}\,|\,\theta)}{f(x_{i'})h(\theta\,|\,x_1,\ldots,x_{i'},\ldots,x_N)g(u'\,|\,\theta)}.
\end{aligned}
$$

Note these terms are defined only where $u'\in\mathcal{X}$ and $i'\in\{1,2,\ldots,N\}$. For

details of the construction of this transition function, see [18].

Thus, the algorithm consists of the following steps performed iteratively:

Algorithm

Given $(x_1, \dots, x_N)$, i, *and* $\theta = (\mu, \Sigma)$ *:*

(1) Put

$$i' = \begin{cases} i+1, \text{ if } i \in \{0, 1, \dots, N-1\} \\ 0, \qquad \text{if } i = N. \end{cases}$$

Generate $Y \sim \mathsf{N}(y \,|\, \mu, \Sigma)$ *if* $i' \in \{1, 2, \dots, N\}$. *Generate* $Y \sim \mathsf{N}(\mu \,|\, \bar{x}, \Sigma/N) \cdot \mathsf{Inv\text{-}W}_{N-1}(\Sigma \,|\, S)$ *if* $i' = 0$.

(2) If $i' \in \{1, 2, \dots, N\}$, *generate* $U \sim \mathsf{U}(0,1)$ *and deliver*

$$x_{i'} = \begin{cases} Y, & \text{if } U \leqslant \alpha_{i'}(x_{i'}, Y) \\ x_{i'}, & \text{otherwise,} \end{cases}$$

where

$$\alpha_{i'}(x_{i'}, y) = \min\{\varrho_{i'}(x_{i'}, y), 1\},$$

with

$$\varrho_{i'}(x_{i'}, y) = \frac{f(y) \cdot \mathsf{N}(\mu \,|\, \bar{x}_y, \Sigma/N) \cdot \mathsf{Inv\text{-}W}_{N-1}(\Sigma \,|\, S_y) \cdot \mathsf{N}(x_{i'} \,|\, \mu, \Sigma)}{f(x_{i'}) \cdot \mathsf{N}(\mu \,|\, \bar{x}, \Sigma/N) \cdot \mathsf{Inv\text{-}W}_{N-1}(\Sigma \,|\, S) \cdot \mathsf{N}(y \,|\, \mu, \Sigma)},$$

$$S = \sum_{n=1}^{N} (x_n - \bar{x})(x_n - \bar{x})^T, \quad \bar{x} = \frac{1}{N} \sum_{n=1}^{N} x_n,$$

$$\begin{aligned} S_Y = {} & (x_1 - \bar{x}_Y)(x_1 - \bar{x}_Y)^T + \dots + (x_{i'-1} - \bar{x}_Y)(x_{i'-1} - \bar{x}_Y)^T \\ & + (Y - \bar{x}_Y)(Y - \bar{x}_Y)^T + (x_{i'+1} - \bar{x}_Y)(x_{i'+1} - \bar{x}_Y)^T + \dots \\ & + (x_N - \bar{x}_Y)(x_N - \bar{x}_Y)^T, \end{aligned}$$

$$\bar{x}_Y = \frac{1}{N}(x_1 + \dots + x_{i'-1} + Y + x_{i'+1} + \dots + x_N).$$

If $i' = 0$, *put* $(\mu, \Sigma) = Y$.

4 A measure of efficiency

Assume for the time being that x is one-dimensional. We shall consider a measure of efficiency (see, e.g., [10,20]) that is based on the asymptotic variance of the sample mean,

$$\bar{x} = \frac{1}{m} \sum_{i=1}^{m} x^{(i)},$$

where $x^{(1)}, \ldots, x^{(m)}$ are samples drawn via a MCMC sampler. Then,

$$\begin{aligned}\mathrm{Var}(\bar{x})\, m &= \sigma^2_{\mathrm{targ}} \left(1 + 2 \sum_{i=1}^{m-1} \Big(1 - \frac{i}{m}\Big) \varrho_i\right) \\ &\approx \sigma^2_{\mathrm{targ}} \left(1 + 2 \sum_{i=1}^{\infty} \varrho_i\right) =: 2\, \sigma^2_{\mathrm{targ}}\, t_{\mathrm{int}},\end{aligned}$$

where σ^2_{targ} is the variance of the target density, ϱ_i is the autocorrelation of the Markov chain at lag i, and t_{int} is the integrated autocorrelation time. To estimate t_{int} we propose the following two approaches.

First,

$$t_{\mathrm{int}} = \frac{1}{2} + \sum_{i=1}^{\infty} \varrho_i \approx \frac{1}{2} + \sum_{i=1}^{K} \varrho_i =: t^{(1)}_{\mathrm{int}},$$

where $K = \min\{i : \varrho_i > 0, \varrho_{i+1} \leqslant 0\}$. In fact, this means that we sum up positive ϱ_i until their values become close to 0. More details on a choice of length K can be found in [6].

Second, it can often be assumed that ϱ_i decays approximately exponentially (see, e.g., [20]). It follows that,

$$|\varrho_i| \sim \exp\Big\{-\frac{i}{t_{\mathrm{exp}}}\Big\},$$

where t_{exp} is the exponential autocorrelation time. In particular, put

$$t_{\mathrm{exp}} = \frac{1}{-\ln|\varrho_1|}.$$

When t_{exp} is large enough, it follows that

$$t_{\mathrm{int}} \approx \sum_{i=0}^{\infty} \exp\Big\{-\frac{i}{t_{\mathrm{exp}}}\Big\} - \frac{1}{2} = \frac{1}{1 - \exp\{-1/t_{\mathrm{exp}}\}} - \frac{1}{2} \approx t_{\mathrm{exp}} =: t^{(2)}_{\mathrm{int}}.$$

We may use t_{int} as a measure of efficiency of a MCMC algorithm. One prefers the algorithm that has the lowest value of t_{int}.

5 Numerical results

Example 1 We consider the following target (see [4]):

$$f(x) = 0.25\, \varphi(x, -6, 2) + 0.7\, \varphi(x, 0, 1) + 0.05\, \varphi(x, 15, 0.1),$$

where

$$\varphi(x, \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}.$$

Suppose

$$g(x) = \sum_{c=1}^{3} w_c \, \varphi(x, \mu_c, \sigma_c^2), \quad \sum_{c=1}^{3} w_c = 1.$$

Put

$$g_0(x) = \frac{1}{3}\,\varphi(x, -10, 4) + \frac{1}{3}\,\varphi(x, 0, 4) + \frac{1}{3}\,\varphi(x, 10, 4).$$

Using CEAIS with $N = 100$, we have obtained:

$$g_{\mathrm{CE}}(x) = 0.195\,\varphi(x, -6.309, 0.870) + 0.775\,\varphi(x, -0.313, 2.144) + 0.030\,\varphi(x, 15.179, 0.194).$$

Using EMAIS with $N = 100$, we have:

$$g_{\mathrm{EM}}(x) = 0.210\,\varphi(x, -6.341, 1.188) + 0.780\,\varphi(x, -0.372, 2.899) + 0.010\,\varphi(x, 15.032, 1.153).$$

Table 1 displays the measure of efficiency for these algorithms.

Table 1
Comparison of CEAIS and EMAIS

Proposal	g_0	g_{CE}	g_{EM}
$t_{\mathrm{int}}^{(1)}$	4.5576	1.2656	2.5536
$t_{\mathrm{int}}^{(2)}$	3.8234	1.0215	1.3235

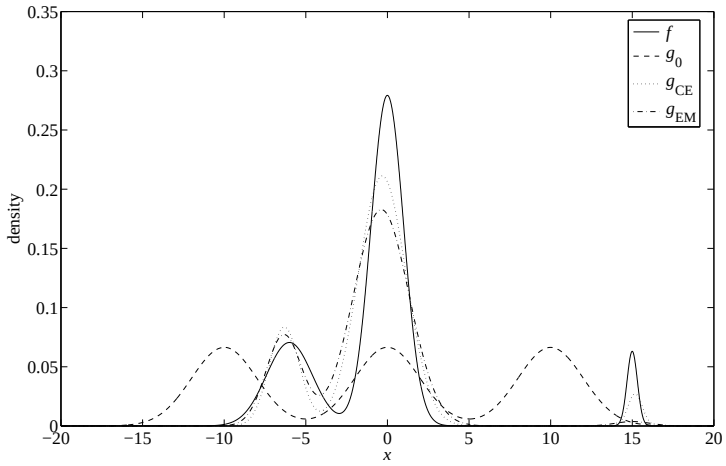

Fig. 1. The plots of the target density $f(x)$ and the proposal densities $g_0(x)$, $g_{\mathrm{CE}}(x)$, $g_{\mathrm{EM}}(x)$

Figure 1 shows the plots of the target density $f(x)$ and the proposal densities $g_0(x)$, $g_{\mathrm{CE}}(x)$, $g_{\mathrm{EM}}(x)$.

In this example, the proposal adopted by CEAIS is somewhat closer to the target than that adopted by EMAIS. This explains why the efficiency of CEAIS substantially improves on the efficiency of EMAIS with respect to both measures of efficiency in Table 1. It may be generally true that CEAIS performs better than EMAIS, because the CE method maximizes the likelihood, whereas the EM method maximizes the expected likelihood.

Example 2 Consider the unimodal target density proportional to

$$\exp\{-x_1^2 - x_2^2 - x_1^4 x_2^4\}.$$

Let us consider the following algorithms: BAIS, the Independent Metropolis-Hastings Algorithm (IMH) with the normal proposal density

$$g(x_1, x_2 \,|\, \delta^2) = \frac{1}{2\pi\delta^2} \exp\Big\{-\frac{x_1^2 + x_2^2}{2\delta^2}\Big\},$$

and the Random Walk Metropolis-Hastings (RWMH) based on the same normal distribution (that is, the Metropolis algorithm with a normal proposal centred on the current element of the chain and having the same covariance matrix). In order to evaluate the measures of efficiency, we shall consider the first component x_1, as suggested in [10]. Figure 2 shows the autocorrelations ϱ_i, where $i = 0, 1, \ldots, 200$ is the lag, for 300 iterations and a burn-in period of 100, sample size $N = 50$, and $\delta^2 = 2$. Note that the autocorrelations first fall below zero at lags of 7, 9 and 16 for the algorithms BAIS, IMH and RWMH respectively. We thus set the value K in our first measure of efficiency $t_{int}^{(1)}$ to $K_{\mathrm{BAIS}} = 6$, $K_{\mathrm{IMH}} = 8$ and $K_{\mathrm{RWMH}} = 15$ respectively. To investigate how $t_{int}^{(1)}$ varies with the choice of K, we plotted Figure 3, which shows the curves of

$$t_{\mathrm{int}}^{(1)}(k) = \frac{1}{2} + \sum_{i=1}^{k} \varrho_i$$

as a function of k for the three algorithms. Note that K_{BAIS}, K_{IMH} and K_{RWMH} correspond to the maxima of the respective curves, which subsequently tend to zero as k increases. The fact that this measure of efficiency is so sensitive to the choice of K makes it seem unappealing, nevertheless it seems clear that choosing K in the above manner results in a sum over terms in which proportional error is minimized. Figure 4 shows the curves of $t_{\mathrm{int}}^{(1)}$, $t_{\mathrm{int}}^{(2)}$ depending on sample size N. Clearly, it is not necessary to use large values of N.

Table 2 displays the measure of efficiency for the three algorithms. Observe that both measures of efficiency are much lower for BAIS than for either IMH or RWMH with the fixed variances shown. To investigate the optimal value of

δ^2 for these algorithms, we plotted Figure 5, showing how the values $t_{\text{int}}^{(1)}$, $t_{\text{int}}^{(2)}$ depend on δ^2 under $N = 50$. We see that IMH has an efficiency comparable to BAIS if we know the value of δ^2 that minimizes some measure of efficiency. At the same time BAIS usually shows better results if we know nothing about the optimal values of parameters of the proposal distribution.

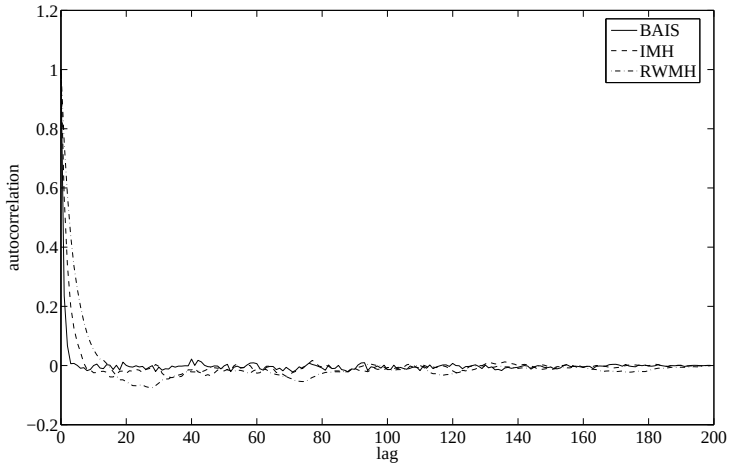

Fig. 2. Curves of autocorrelations

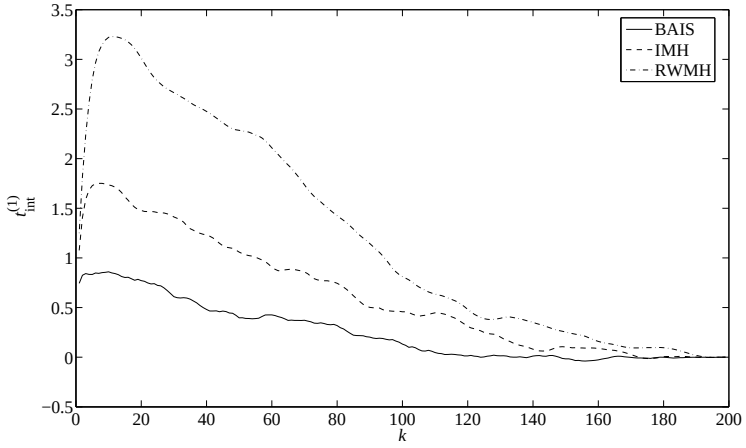

Fig. 3. Curves of $t_{\text{int}}^{(1)}$

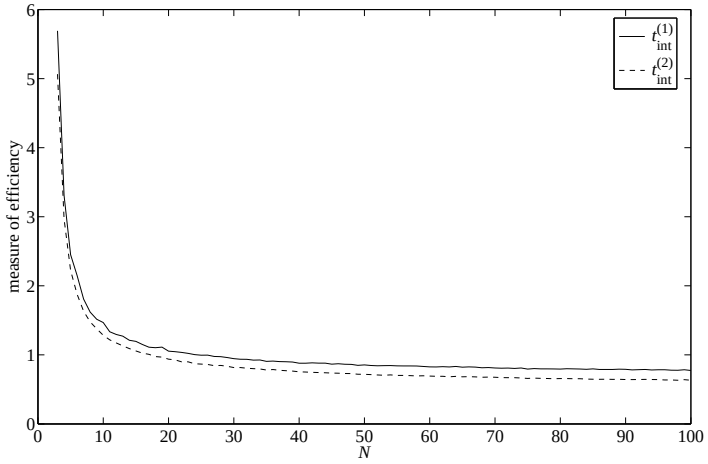

Fig. 4. The curves of $t_{\text{int}}^{(1)}$, $t_{\text{int}}^{(2)}$ for BAIS depending on sample size N

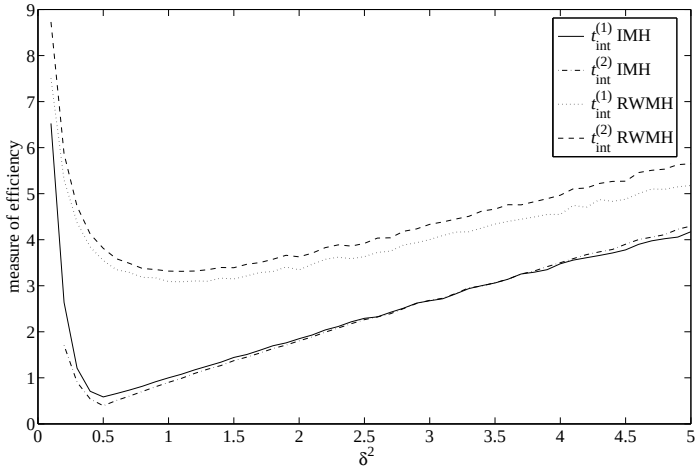

Fig. 5. The curves of $t_{\text{int}}^{(1)}$, $t_{\text{int}}^{(2)}$ depending on δ^2 under $N = 50$

Table 2
Comparison of MCMC algorithms

Algorithm	BAIS	IMH		RWMH	
		$\delta^2 = 2$	$\delta^2 = 4$	$\delta^2 = 2$	$\delta^2 = 4$
$t_{\text{int}}^{(1)}$	0.8166	1.8632	3.4111	3.5337	4.5746
$t_{\text{int}}^{(2)}$	0.6926	1.7650	3.4784	3.5582	4.9604

Since the measures of efficiency for BAIS are less than for IMH and for RWMH, it appears that BAIS is preferable to either of these samplers for this distribution. We speculate that this will generally be true for unimodal distributions.

Example 3 Consider the bimodal target density proportional to

$$\exp\{-(x_1^2x_2^2 + x_1^2 + x_2^2 - 8x_1 - 8x_2)/2\}.$$

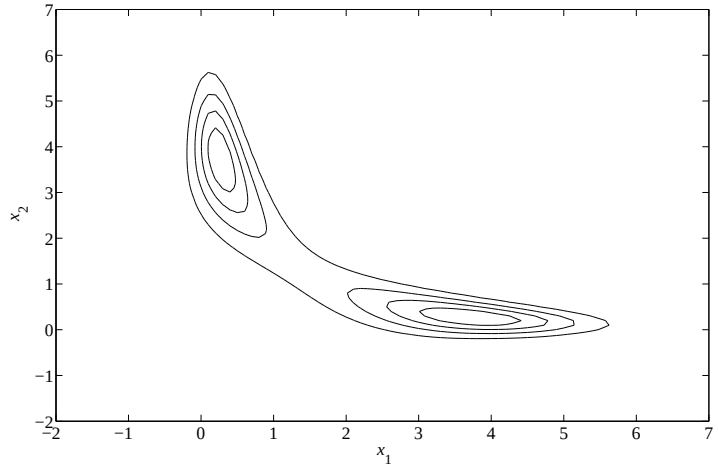

Fig. 6. The contour plot of the bimodal target density

Figure 6 shows the contour plot of the target density.

Let us consider the following algorithms: BAIS, IMH with the normal proposal density

$$g(x_1, x_2 \,|\, \delta^2) = \frac{1}{2\pi\delta^2} \exp\Big\{-\frac{x_1^2 + x_2^2}{2\delta^2}\Big\},$$

RWMH based on the same normal distribution, and CEAIS with the following proposal density

$$g(x_1, x_2) = \sum_{c=1}^{2} w_c \, \varphi(x_1, x_2, \mu_c, \Sigma_c), \quad w_1 + w_2 = 1,$$

where $\varphi(x_1, x_2, \mu_c, \Sigma_c)$ is a bivariate normal density. Using pre-runs and CEAIS we get estimators of the parameters:

$$w_1 = 0.5621,\ w_2 = 0.4379,\ \mu_1 = \begin{pmatrix} 0.4541 \\ 3.2189 \end{pmatrix},\ \mu_2 = \begin{pmatrix} 3.3046 \\ 0.4943 \end{pmatrix},$$

$$\Sigma_1 = \begin{pmatrix} 0.3937 & -0.6118 \\ -0.6118 & 1.7682 \end{pmatrix},\ \Sigma_2 = \begin{pmatrix} 2.0205 & -0.7315 \\ -0.7315 & 0.4631 \end{pmatrix}.$$

As before, we only consider the first component x_1. Figure 7 shows the curves of autocorrelations ϱ_i, for lags $i = 0, 1, \ldots, 200$, with sample size $N = 50$. We ran the algorithms for 300 iterations, using 100 as a burn-in period.

Table 3 displays the measures of efficiency for these algorithms. We see that the measures of efficiency for CEAIS are less than for the other algorithms, and that BAIS also performs well in comparison to the other methods.

Table 3
Comparison of MCMC algorithms

Algorithm	BAIS	IMH		RWMH		CEAIS
		$\delta^2 = 2$	$\delta^2 = 4$	$\delta^2 = 2$	$\delta^2 = 4$	
$t_{\text{int}}^{(1)}$	3.8039	15.8197	12.1685	13.2056	12.5742	1.3776
$t_{\text{int}}^{(2)}$	3.4903	13.0056	13.2340	14.2064	15.3989	1.2259

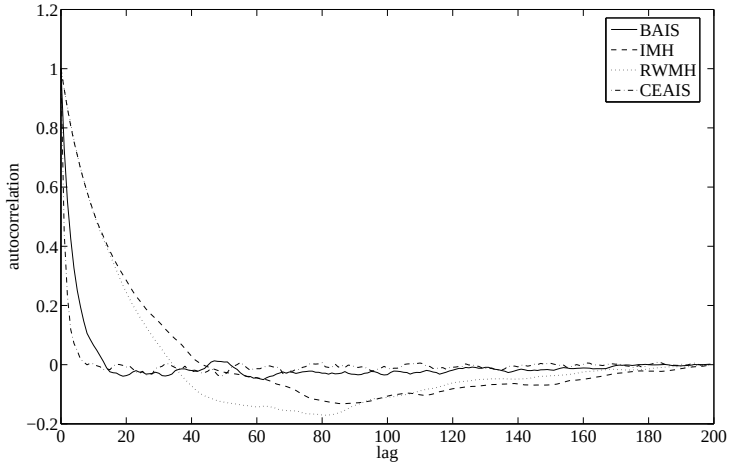

Fig. 7. Curves of autocorrelations

This example illustrates that if the target density has more than one mode, CEAIS with a proposal mixture-model (which parameters were estimated by using pre-runs) is more efficient than other algorithms with a unimodal proposal density. It also illustrates that BAIS can perform quite efficiently for a bimodal distribution, even with a unimodal proposal.

Example 4 In this example, we apply BAIS to an important problem that arises in Comparative Genomics. Our intent is to demonstrate the applicability of the method to problems of genuine significance. Specifically, the problem is to estimate the proportion of a given genome that is conserved in two lineages. One way in which this problem is approached is to first align the two genomes and then perform a "sliding window" analysis — a form of Loess analysis. The alignment is divided into contiguous segments of a fixed width or "window length" and the proportion of matches is determined for each window. Each window thus produces a single data point in the interval [0,1]. The proportion of the genome that is conserved can then be estimated by fitting a

mixture model to these data points and estimating the mixture proportion of the most slowly evolving component. This approach has been used to estimate the proportion of the human genome that is conserved in mice to be about 5% [30]. Here we fit a three component model to data obtained by aligning the two fruit fly species Drosophila melanogaster and Drosophila simulans, using 200 as the window length.

Let $y = (y_1, \ldots, y_M) \in [0,1]^M$ be given data. Denote $\theta = (\pi_1, \pi_2, \alpha_1, \beta_1, \alpha_2, \beta_2, \alpha_3, \beta_3)$, $\pi_1 \geqslant 0$, $\pi_2 \geqslant 0$, $\pi_1 + \pi_2 \leqslant 1$, $\alpha_i > 0$, $\beta_i > 0$, $i = 1, 2, 3$. Put

$$h(x \,|\, \theta) = \pi_1 \mathsf{B}(x \,|\, \alpha_1, \beta_1) + \pi_2 \mathsf{B}(x \,|\, \alpha_2, \beta_2) + (1 - \pi_1 - \pi_2)\mathsf{B}(x \,|\, \alpha_3, \beta_3),$$

where $\mathsf{B}(\cdot \,|\, \alpha, \beta)$ is a Beta density. Suppose that (π_1, π_2) is uniformly distributed, and parameters $\alpha_i > 0$, $\beta_i > 0$, $i = 1, 2, 3$, are exponentially distributed. Then the target

$$f(\theta \,|\, y) \propto \prod_{m=1}^{M} h(y_m \,|\, \theta) \exp\{-\alpha_1 - \beta_1 - \alpha_2 - \beta_2 - \alpha_3 - \beta_3\}.$$

Using BAIS we consider an 8-dimensional normal proposal distribution. We start with

$$\mu = (0.33, 0.33, 100, 100, 100, 100, 100, 100)^T,$$
$$\Sigma = (0.33, 0.33, 50, 50, 50, 50, 50, 50) \cdot I_8,$$

where I_8 is the 8×8 identity matrix. In fact, the limiting distribution is independent of the initial values of μ and Σ, and they should not greatly affect the samplers efficiency. We ran the algorithm for 7000 iterations, using $N = 50$ chains. We have obtained the final value

$$\hat{\theta} = (0.4307, 0.0081, 2.5950, 30.2073, 28.7872, 43.4301, 2.5360, 71.0390).$$

Figure 8 shows the three components of the mixture, normalized to π_1, π_2, and $\pi_3 = 1 - \pi_1 - \pi_2$ respectively. Figure 9 shows the frequency histogram and the density $h(x \,|\, \hat{\theta})$. Figure 10 plots the log-likelihood values for 7000 iterations of the algorithm, showing rapid convergence to an optimal value.

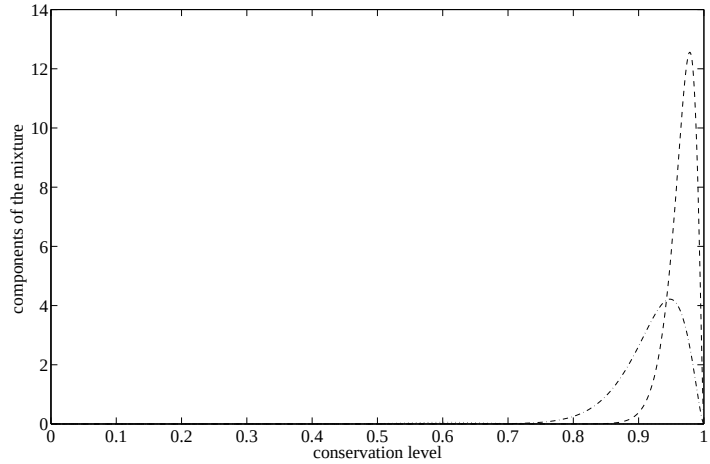

Fig. 8. The three components of the mixture, normalized to π_1, π_2, and π_3 respectively. Note that one of the groups has a mixture proportion of only 0.0081 and is only barely visible in the figure.

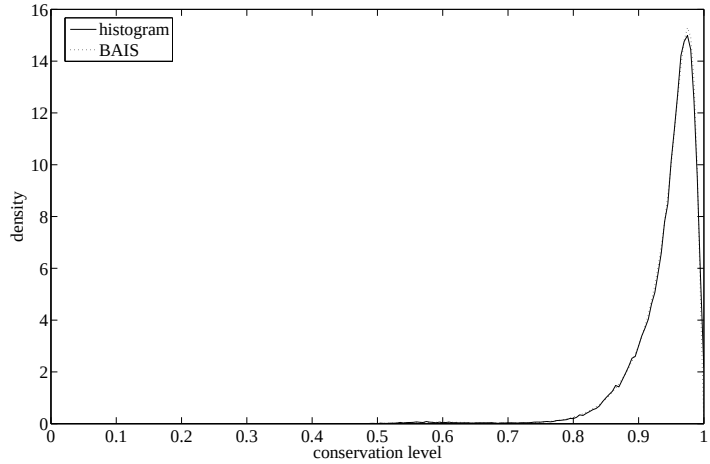

Fig. 9. The frequency histogram and the curve obtained by BAIS

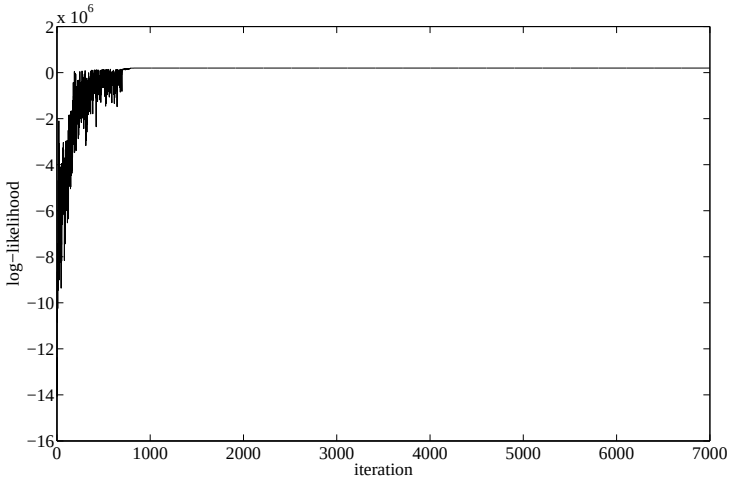

Fig. 10. The curve of log-likelihood

For the purpose of this example, we assume that the conserved fraction of the genome corresponds to the most slowly evolving component of the mixture model. The mixture proportion of this component is 0.5612. Thus approximately 56% of the aligned portion of the D. melanogaster genome is conserved. This figure is concordant with an imprecise estimate obtained by Andolfatto [2] via an entirely different approach. It is somewhat less than, but still comparable to, a currently unpublished estimate obtained by author Keith using a much more difficult and computationally intensive segmentation method.

6 Conclusion

In this paper we have introduced two new adaptive MCMC methods called Adaptive Independence Samplers. The examples reported in the paper show that the samplers can be superior to classical Metropolis-Hastings algorithms such as the Independent Metropolis-Hastings algorithm and the Random Walk Metropolis-Hastings Algorithm in terms of efficiency and convergence properties. The new methods are easy to implement.

In the first approach (CEAIS), where we use pre-runs, the adaptation takes place only in an initial phase and afterwards the proposal density is fixed. Therefore the convergence is ensured by the basic theory. One advantage of this approach is the convenient way in which update formulae can be obtained for the adaptive phase. Another advantage of this approach is that it lends itself easily to the use of a mixture model proposal density. Unsurprisingly, we found that the AIS with a mixture model proposal is more effecient than unimodal proposals in the case of a multimodal target density.

In the second, Bayesian, approach (BAIS) we have proved, using the Generalized Markov Sampler, that the target distribution is the limiting distribution of the corresponding process. This is an important feature, as it means that the adaptive phase can be continued indefinitely, and the user does not have to decide at what point to terminate the adaptive phase. However, the current implementation of BAIS does not easily accomodate a mixture model proposal in the way that CEAIS does. Thus it has some of the same disadvantages as other methods with unimodal proposals when attempting to sample from a multimodal target density. Nevertheless, we found that BAIS performed well even for a bimodal distribution.

A possible drawback of our algorithms is that for certain problems the computational cost of performing pre-runs in CEAIS or of running multiple chains in BAIS may offset the benefits of accelerated convergence and mixing. However, this is an issue shared with all adaptive MCMC algorithms. Adaptive methods are not appropriate for such problems, and the MCMC practitioner must use his or her judgement to decide when adaptive methods will be advantageous. General rules for deciding when adaptive techniques should be implemented are a substantial matter for further research.

The strength of adaptive MCMC algorithms lies in the fact that parameters of the proposal distribution can be updated to improve the efficiency and convergence of the sampler, whereas the classical MCMC algorithms have the parameters of the proposal fixed. In this paper, we have described two new adaptive MCMC methods based on the Independence Sampler, both of which provide simple, natural, efficient and easily implemented procedures for updating the proposal.

Acknowledgement

The authors are grateful to Prof. Kerrie Mengersen for comments on the manuscript.

References

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[2] P. Andolfatto. Adaptive evolution of non-coding dna in drosophila. *Nature*, 437:1149–1152, 2005.

[3] Y. F. Atchade and J. S. Rosenthal. On adaptive Markov chain Monte Carlo algorithms. *Bernoulli*, 11:815–828, 2005.

[4] D. Chauveau and P. Vandekerkhove. Improving convergence of the Hastings-Metropolis algorithm with an adaptive proposal. *Scandinavian Journal of Statistics*, 29(1):13–29, 2002.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[6] G. S. Fishman. *Monte Carlo: Concepts, Algorithms and Aplications*. Springer, New York, 1996.

[7] J. Gasemyr. On an adaptive version of the MetropolisHastings algorithm with independent proposal distribution. *Scandinavian Journal of Statistics*, 30(1):159–173, 2003.

[8] A. E. Gelfand and S. K. Sahu. On Markov chain Monte Carlo acceleration. *Journal of Computational and Graphical Statistics*, 3(3):261–276, 1994.

[9] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, London, second edition, 2003.

[10] A. G. Gelman, G. O. Roberts, and W. R. Gilks. Efficient metropolis jumping rules. In J. M. Bernardo, J. O. Berger, A. F. David, and A. F. M. Smith, editors, *Bayesian Statistics V*, pages 599–608. Oxford Univ. Press, New York, 1996.

[11] W. Gilks, G. Roberts, and E. George. Adaptive direction sampling. *The Statistician*, 43(1):179–189, 1994.

[12] W. Gilks, G. Roberts, and S. Sahu. Adaptive Markov chain Monte Carlo through regeneration. *Journal of the American Statistical Association*, 93:1045–1054, 1998.

[13] H. Haario, E. Saksman, and J. Tamminen. Adaptive proposal distribution for random walk Metropolis algorithm. *Computational Statistics*, 14:375–395, 1999.

[14] H. Haario, E. Saksman, and J. Tamminen. An adaptive Metropolis algorithm. *Bernoulli*, 7(2):223–242, 2001.

[15] H. Haario, E. Saksman, and J. Tamminen. Componentwise adaptation for high dimensional MCMC. *Computational Statistics*, 20:265–273, 2005.

[16] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[17] L. Holden. Adaptive chains. Technical report, Norwegian Computing Centre, P. O. Box 114 Blindern, N-0314, Oslo, Norway, 2000.

[18] J. Keith, D. P. Kroese, and D. Bryant. A generalized Markov sampler. *Methodology and Computing in Applied Probability*, 6(1):29–53, 2004.

[19] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann.Math.Stat.*, 22:79–86, 1951.

[20] J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer-Verlag, 2001.

[21] G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. John Wiley & Sons, New York, 1997.

[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, and A. H. Teller. Equations of state calculations by fast computing mashines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[23] P. Mykland, L. Tierney, and B. Yu. Regeneration in Markov chain samplers. *Journal of the American Statistical Association*, 90:233–241, 1995.

[24] C. Pasarica and A. Gelman. Adaptively scaling the Metropolis algorithm using expected squared jumped distance. Technical report, Department of Statistics, Columbia University, 2003.

[25] R. Y. Rubinstein and D. P. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer-Verlag, New York, 2004.

[26] S. K. Sahu and A. A. Zhigljavsky. Self regenerative Markov chain Monte Carlo with adaptation. *Bernoulli*, 9:395–422, 2003.

[27] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[28] L. Tierney and A. Mira. Some adaptive Monte Carlo methods for Bayesian inference. *Statistics in Medicine*, 18:2507–2515, 1999.

[29] G. R. Warnes. The normal kernel coupler: An adaptive Markov chain Monte Carlo method for efficiently sampling from multi-modal distributions. Technical Report 39, Department of Statistics, University of Washington, 2003.

[30] R. H. Waterston, K. Lindblad-Toh, E. Birney, J. Rogers, J.F. Abril, and et al. Initial sequencing and comparative analysis of the mouse genome. *Nature*, 420:520–562, 2002.