

# On Estimating Non-standard Discrete Distributions Using Adaptive MCMC Methods

Hatem Baffoun<sup>1</sup>, Mekki Hajlaoui<sup>2</sup> & Abdeljelil Farhat<sup>1</sup>

<sup>1</sup> Unité de recherche "Economie Appliquée & Simulation", Faculté des Sciences Economiques et de Gestion de Mahdia, Sidi Messoud 5111, Tunisie-Tunisia

<sup>2</sup> QUantitative Analysis Research Group (QUARG), Institut des Hautes Etudes Commerciales de Carthage, Carthage-Présidence 2016, Tunisie-Tunisia

Correspondence: Faculté des Sciences Economiques et de Gestion de Mahdia, Sidi Messoud 5111, Tunisie-Tunisia.  
E-mail: hatem.baffoun@yahoo.fr

Received: January 22, 2018 Accepted: February 5, 2018 Online Published: February 28, 2018

doi:10.5539/ijsp.v7n3p1 URL: <https://doi.org/10.5539/ijsp.v7n3p1>

## Abstract

In this paper, we compare empirically the performance of some adaptive MCMC methods, that is, Adaptive Metropolis (AM) algorithm, Single Component Adaptive Metropolis (SCAM) algorithm and Delayed Rejection Adaptive Metropolis (DRAM) algorithm. The context is the simulation of non-standard discrete distributions. The performance criterion used is the precision of the frequency estimator. An application to a Bayesian hypothesis testing problem shows the superiority of the DRAM algorithm over the other considered sampling schemes.

**Keywords:** Adaptive MCMC methods, discrete probability distribution, frequency estimator, acceptance rate, Bayesian inference

## 1. Introduction

The principle of Markov chain Monte-Carlo (MCMC) approach is to construct an ergodic Markov chain whose stationary distribution corresponds exactly to the simulating density. To do this, several sampling schemes have been proposed and the best known one is the Metropolis-Hastings (M-H) algorithm, see (Hastings, 1970). This algorithm uses an acceptance-rejection mechanism on values generated from some proposal distribution. Its main advantage is that the convergence is guaranteed whatever the used proposal density. But, a bad choice of this distribution may be disastrous and leads to a very slow convergence. This is often the case when the density of interest is complicated or of high dimension.

To overcome this limit, diverse adaptive versions of the M-H algorithm are proposed in the literature [for an overview, see (Bardenet, Cappé, Fort & Kégl, 2015), (Liang, Liu & Carroll, 2010) and (Vihola, 2012)]. These methods allow to adjust, automatically, the parameters of the proposal distribution in order to speed up the convergence. However, most of them seem to be problematic due to the difficulty in ensuring the ergodicity of the simulated chains. Consequently, the theoretical convergence results of M-H algorithm will not be applicable.

The Adaptive Metropolis (AM) algorithm is one of the most successfully applied adaptive methods. Developed originally by (Haario, Saksman & Tamminen, 2001), this sampling scheme has the advantage to preserve the correct ergodicity properties [see (Saksman & Vihola, 2010)], which ensures the convergence of the chain to the target distribution. Its basic principle is to use multi-normal proposal density whose covariance matrix is tuned at each iteration. Recently, the AM algorithm has aroused the interest of many scientists and new variants had been proposed. Thus, to improve the performance of this sampler, particularly for high dimensions, (Haario, Saksman & Tamminen, 2005) have developed the Single Component Adaptive Metropolis (SCAM) algorithm. Likewise, (Haario, Laine, Mira & Saksman, 2006) have proposed the Delayed Rejection Adaptive Metropolis (DRAM) algorithm which combines both the AM sampler and the Metropolis-Hastings algorithm with delayed rejection (MHDR).

In the literature, the most frequently used criteria to test the efficiency of MCMC methods are both convergence rate and accuracy of considered estimators. Although many works had revealed the rapid convergence of M-H adaptive algorithms, few studies focused on the assessment of these methods in terms of the accuracy of the calculated estimators.

In this article, we suggest to compare and analyze, experimentally, the performances of AM, SCAM and DRAM algorithms, in the context of simulating non-standard discrete distributions. The performance criterion considered is the precision of the frequency estimator. Hence, the remaining part of this work is organized as follows. In section 2, we explain the details of the AM sampler. Section 3 contains a description of the SCAM algorithm. In section 4, the foundations of the DRAM method are given with its main characteristics. Finally, our empirical study is presented in section 5,

where the results are analyzed and discussed.

## 2. AM algorithm

To properly set up the M-H algorithm, it is crucial to well choose the parameters of the proposal distribution and particularly the variance. Indeed, if the variance is weak, the acceptance rate will be high but the chain will converge very slowly because the steps are very small. In case of the opposite sense, if the variance is large, the algorithm will reject most of the proposed candidates and, consequently, the speed of its convergence will be weak. In general, the adjustment of the proposal variance is heuristic and it forms a quite tricky problem.

The AM algorithm aims to palliate this particular difficulty. The idea is to modify, at each iteration, the covariance matrix of the proposal distribution in order to correspond it to the target density. Just one obstacle is that the target covariance matrix is unknown in general. In order to get around this problem, the AM algorithm estimates on each iteration the target covariance matrix from the past values of the simulated chain.

Suppose that the AM algorithm is used to simulate a target density  $\pi$  and  $(x_t, t = 0, 1, \dots)$  specifies the obtained Markov chain. The first step is to fix out the initial covariance matrix  $C_0$ . On the  $(t + 1)^{th}$  iteration of the algorithm, a candidate is generated from a multi-normal proposal distribution, centered at the current state  $x_t$  and with covariance matrix  $C_{t+1}$  defined by

$$C_{t+1} = \begin{cases} C_0, & \text{if } t + 1 \leq t_0, \\ s_d \text{COV}(x_0, \dots, x_t) + s_d \epsilon I_d, & \text{if } t + 1 > t_0. \end{cases} \quad (1)$$

The term  $t_0$  indicates the time from which the proposal covariance matrix is adjusted, the parameter  $s_d$  is a scaling factor that depends to the dimension  $d$  of the states space, the term  $\epsilon$  insures the non-singularity of the matrix  $C_{t+1}$  and  $I_d$  denotes the identity matrix of order  $d$ .

Similarly to the M-H sampling scheme, the AM algorithm uses an acceptance/rejection mechanism to select proposed candidates through the following steps:

1. Set  $x_0, C_0, t_0, s_d$  and  $\epsilon$ .
2. On the  $(t + 1)^{th}$  iteration:
  - (a) determine  $C_{t+1}$  using formula (1).
  - (b) generate a candidate  $y$  from the proposal distribution  $N_d(x_t, C_{t+1})$ .
  - (c) compute the acceptance probability

$$\alpha(x_t, y) = \min \left[ 1, \frac{\pi(y)}{\pi(x_t)} \right].$$

- (d) take

$$x_{t+1} = \begin{cases} y & \text{with probability } \alpha(x_t, y), \\ x_t & \text{with probability } 1 - \alpha(x_t, y). \end{cases}$$

3. Set  $t \leftarrow t + 1$  and go to step 2.

It is important to note that the process  $(x_t, t = 0, 1, \dots)$  is far to be markovien since its future development depends on the whole history. But its ergodicity was rigorously proved by (Haario, Saksman & Tamminen, 2001).

## 3. SCAM Algorithm

In the literature, the AM algorithm has been applied with success for various problems and its advantages are identified by many authors. However, it is shown by (Haario, Saksman & Tamminen, 2005) that this simulation scheme loses efficiency in high dimension problems. In this particular case, adjusting the covariance matrix leads to very complicated calculations that slow down the convergence of the algorithm.

To overcome these difficulties, (Haario, Saksman & Tamminen, 2005) have suggested a new version of the AM algorithm named the SCAM algorithm. Its special feature is that the chain future state, represented by a vector, is generated component by component using univariate proposal densities. These distributions are normal and whose variance is adjusted at each iteration similarly to the AM method. In this way, we pass from simulating a multivariate variable to simulating many univariate variables which is often an easier task.

Let  $\pi$  be a  $d$ -dimensional distribution, that we desire to simulate through the SCAM algorithm. Suppose  $(x_t, t = 0, 1, \dots)$  is the produced markov chain and  $x_t^1, \dots, x_t^d$  are the  $d$  components of the vector  $x_t$ . On the  $(t + 1)^{th}$  iteration of the

algorithm and to simulate the  $t^{th}$  coordinate of  $x_{t+1}$ , we generate a candidate from the normal distribution, centered at  $x_t^i$ , and with variance  $C_{t+1}^i$  such that

$$C_{t+1}^i = \begin{cases} C_0^i, & \text{if } t + 1 \leq t_0, \\ s_d \text{COV}(x_0^i, \dots, x_t^i) + s\epsilon, & \text{if } t + 1 > t_0. \end{cases} \tag{2}$$

where  $C_0^i$  is the initial value of the variance,  $t_0$  is the moment at which the variance is adjusted,  $s$  is a scaling factor whose value exceeds 1 and  $\epsilon$  is a positive constant such that  $0 < \epsilon < 1$ .

In order to accept/reject the proposed candidate, SCAM algorithm uses the M-H dynamic. Thus, starting from  $x_t^i$ , the component  $x_{t+1}^i$  is obtained by the following scheme:

1. Determine  $C_{t+1}^i$  using the formula (2).
2. Generate a candidate  $y^i$  from the proposal distribution  $N(x_t^i, C_{t+1}^i)$ .
3. Compute the acceptance probability

$$\alpha(x_t^i, y^i) = \min \left[ 1, \frac{\pi(x_{t+1}^1, \dots, x_{t+1}^{i-1}, y^i, x_t^{i+1}, \dots, x_t^d)}{\pi(x_{t+1}^1, \dots, x_{t+1}^{i-1}, x_t^i, x_t^{i+1}, \dots, x_t^d)} \right].$$

4. Take

$$x_{t+1}^i = \begin{cases} y^i & \text{with probability } \alpha(x_t^i, y^i), \\ x_t^i & \text{with probability } 1 - \alpha(x_t^i, y^i). \end{cases}$$

As for the AM sampler, the generated process  $(x_t, t = 0, 1, \dots)$  is no more a Markov chain. However and according to (Haario, Saksman & Tamminen, 2005), this process meets the ergodicity property that ensures its convergence to the distribution of interest  $\pi$ .

**4. DRAM Algorithm**

Originally developed by (Haario, Laine, Mira & Saksman, 2006), the DRAM sampling scheme aims to combine the rapidity of the AM algorithm with the high performance of the MHDR algorithm. The latter was suggested by (Mira, 1998) and constitutes an improved version of the M-H algorithm whose aim is to reduce the variance of the computed estimators. The idea is to propose more than one candidate within the same iteration in order to increase the acceptance rate, see (Baffoun, Hajlaoui & Farhat, 2017).

**4.1 MHDR Algorithm**

Suppose that we desire to construct a Markov chain  $(x_t, t = 0, 1, \dots)$  whose invariant distribution is exactly  $\pi$ . Analogously to the M-H algorithm, on the  $(t + 1)^{th}$  iteration of MHDR algorithm, the first step is to generate a candidate  $y_1$  from the proposal distribution  $q_1(x_t, \cdot)$ . This candidate is accepted as a new state of the chain with the probability

$$\alpha_1(x_t, y_1) = \min \left[ 1, \frac{\pi(y_1)q_1(y_1, x_t)}{\pi(x_t)q_1(x_t, y_1)} \right].$$

In case of rejection and instead of staying in the current state, as suggested by M-H algorithm, we pass to step two by generating a new candidate  $y_2$  from the proposal density  $q_2(x_t, y_1, \cdot)$ . This candidate is accepted with the probability

$$\alpha_2(x_t, y_2) = \min \left[ 1, \frac{\pi(y_2)q_1(y_2, y_1)q_2(y_2, y_1, x_t)[1 - \alpha_1(y_2, y_1)]}{\pi(x_t)q_1(x_t, y_1)q_2(x_t, y_1, y_2)[1 - \alpha_1(x_t, y_1)]} \right].$$

This sampler can be applied with fixed or random number of steps by iteration, see (Mira, 1998). It is interesting to note that all acceptance probabilities are computed so that reversibility of the chain, with respect to the density of interest, is preserved separately at each step.

**4.2 Combining MHDR and AM Algorithms**

The DRAM sampling scheme is based on a similar concept as MHDR method. Except that the candidate of the first step is now generated by AM algorithm rather than by M-H algorithm. For this fact, a normal proposal distribution is used which is centered on the current value of the chain and whose variance is adjusted at each iteration. For the second step,

(Haario, Laine, Mira & Saksman, 2006) recommend that the proposal density of the previous step is maintained while reducing its variance.

Let  $\pi$  be a  $d$ -dimensional distribution that we desire to simulate through the DRAM algorithm. On the  $(t + 1)^{th}$  iteration of the algorithm, we use, for the first step, the proposal density  $q_1 = N_d(x_t, C_{t+1})$  where  $C_{t+1}$  specifies the covariance matrix computed according to eq.(1). The proposal density of the second step is  $q_2 = N_d(x_t, \alpha C_{t+1})$  with  $\alpha$  is a scaling factor chosen from the interval ]0, 1[. Thus, the transition from the current state  $x_t$  to the future state  $x_{t+1}$  is done as follows:

1. Set the initial state  $x_0$  and the initial covariance matrix  $C_0$ .

2. On the  $(t + 1)^{th}$  iteration:

(a) determine  $C_{t+1}$ .

(b) generate a candidate  $y_1$  from the proposal distribution  $q_1$ .

(c) compute the acceptance probability

$$\alpha_1(x_t, y_1) = \min \left[ 1, \frac{\pi(y_1)}{\pi(x_t)} \right].$$

(d) set  $x_{t+1} = y_1$  with probability  $\alpha_1(x_t, y_1)$ , otherwise

i. generate a candidate  $y_2$  from the proposal density  $q_2$ ,

ii. calculate the acceptance probability

$$\alpha_2(x_t, y_2) = \min \left[ 1, \frac{\pi(y_2)q_1(y_2, y_1)[1 - \alpha_1(y_2, y_1)]}{\pi(x_t)q_1(x_t, y_1)[1 - \alpha_1(x_t, y_1)]} \right].$$

iii. take

$$x_{t+1} = \begin{cases} y_2 & \text{with probability } \alpha_2(x_t, y_2), \\ x_t & \text{with probability } 1 - \alpha_2(x_t, y_2). \end{cases}$$

3. Set  $t \leftarrow t + 1$  and go to step (2).

As for the AM algorithm, the simulated process  $(x_t, t = 0, 1, \dots)$  does not form a Markov chain since its future development depends to all its past states. However and according to (Haario, Laine, Mira & Saksman, 2006), this process complies with the ergodicity property that ensures its convergence to the distribution of interest  $\pi$ .

### 5. Experimental Study

This section aims to compare, empirically, the performances of AM, SCAM, and DRAM algorithms. For this purpose, we use a real application from (Liang & Liu, 2005). It is a Bayesian hypothesis testing problem which consists of checking if one of the coefficients of a logistic regression model is positive or not.

#### 5.1 Data Description

In this study, the dataset is taken from (Ashford, 1959). It concerns the proportion of British coal miners who present symptoms of severe pneumoconiosis and the length of service in the colliery. Following (Liang & Liu, 2005), we consider the logistic regression model given by

$$E(y_i) = \frac{\exp(\beta_0 + \beta_1 z_i + \beta_2 z_i^2)}{1 + \exp(\beta_0 + \beta_1 z_i + \beta_2 z_i^2)}, \quad i = 1, \dots, 371,$$

where  $\beta_0, \beta_1$ , and  $\beta_2$  are the regression coefficients,  $z_i$  is the number of years of service of the  $i^{th}$  miner and  $y_i$  is the binary indicator of severe pneumoconiosis. By assuming that  $\beta_0, \beta_1$ , and  $\beta_2$  are independently distributed with a common normal density of mean 0 and variance  $\sigma^2 = 100$  and if we let  $\beta$  be the column vector of these coefficients, the log-posterior density of  $\beta$  is defined (up to an additive constant) by

$$\log \pi(\beta|D) = \sum_{i=1}^{371} y_i(\beta_0 + \beta_1 z_i + \beta_2 z_i^2) - \sum_{i=1}^{371} \log(1 + \exp(\beta_0 + \beta_1 z_i + \beta_2 z_i^2)) - \frac{1}{2\sigma^2}(\beta_0^2 + \beta_1^2 + \beta_2^2),$$

where  $D$  is the set of all observations.

The aim of this study is to test the null hypothesis  $\beta_2 < 0$ . In (Liang & Liu, 2005), this problem is reduced to the comparison of the posterior probabilities  $\pi(A_1)$  and  $\pi(A_2)$ , where  $A_1 = \{\beta : \beta_2 < 0\}$  and  $A_2 = \{\beta : \beta_2 \geq 0\}$ . The solution

consists of sampling at first, through a MCMC algorithm, an ergodic chain  $(x_t, t = 0, 1, \dots)$  whose invariant distribution is the posterior  $\pi(\beta|D)$  to approximate later  $\pi(A_i), i = 1, 2$ , with the frequency estimator,

$$\hat{\pi}^f(A_i) = \frac{1}{n} \sum_{t=0}^{n-1} I(x_t \in A_i), i = 1, 2, \tag{3}$$

where  $n$  is the number of iterations performed and  $I(\cdot)$  is the indicator function of  $A_i$ . In this study, we suggest to resolve this problem in a similar manner using AM, SCAM and DRAM algorithms. The objective is to compare the performances of these adaptive MCMC methods in terms of the precision of the frequency estimator.

### 5.2 Implementation

To set up a MCMC method, it is important to fix out some parameters that largely affect the simulation results. The choice of these parameters is based mainly on the characteristics of the considered problem. In the sequel, we detail our implementation for each considered sampling scheme.

#### 5.2.1 Parameterizing of the AM Algorithm

Contrary to the M-H algorithm where the choice of the proposal distribution is arbitrary, the AM sampler uses a multi-normal proposal density. The proper application of this method requires optimization of the proposal covariance matrix in order to better approximate the target covariance matrix.

As explained in section 2, AM algorithm generates the first  $t_0$  observations from a fixed candidate density, centered at the current state of the chain and with covariance matrix  $C_0$ . (Haario, Saksman & Tamminen, 2001) recommend that this first part of the simulated chain should be relatively short in order to speed up the convergence of the algorithm. For this application, we choose  $t_0 = 1000$  and  $C_0 = \text{diag}[1, 0.001, 0.0001]$ . Then, to construct the rest of the chain, the AM sampler uses a proposal distribution whose covariance matrix is updated at each iteration according to eq.(1). The calculation of the latter requires to set the parameters  $s_d$  and  $\epsilon$ . Following (Haario, Saksman & Tamminen, 2001), we take the scaling parameter to be  $s_d = \frac{(2.4)^2}{d}$ . Concerning the parameter  $\epsilon$ , which is chosen to be small (compared to the dimension of the states space), we assign it a value of 0.001.

To simulate from  $\pi(\beta|D)$ , we choose  $\beta_0$ , the null column vector of order 3, as the starting point of the AM algorithm. On  $(t + 1)^{th}$  iteration, a value  $y$  is generated from the candidate distribution  $N_3(\beta_t, C_{t+1})$  where  $\beta_t$  is the current state of the chain and  $C_{t+1}$  is the covariance matrix defined by

$$C_{t+1} = \begin{cases} C_0, & \text{if } t + 1 \leq 1000, \\ \frac{5.760}{3} \text{cov}(\beta_0, \dots, \beta_t) + \frac{0.006}{3} I_3, & \text{if } t + 1 > 1000. \end{cases}$$

The value  $y$  is then accepted with the probability

$$\alpha(\beta_t, y) = \min \left[ 1, \frac{\pi(y)}{\pi(\beta_t)} \right].$$

In case of rejection, the produced chain stays in its current state i.e.  $\beta_{t+1} = \beta_t$ .

#### 5.2.2 Parameterizing of the SCAM Algorithm

Recall that in case of the SCAM algorithm, the realizations are simulated component by component using proposal normal distributions. To generate a Markov chain with stationary distribution  $\pi(\beta|D)$ , each iteration of this sampler scheme contains three steps. Given the current state of the chain  $\beta_t = (\beta_0^t, \beta_1^t, \beta_2^t)'$ , the first step consists of replacing the first coordinate  $\beta_0^t$  by  $\beta_0^{t+1}$  to thereby produce the vector  $(\beta_0^{t+1}, \beta_1^t, \beta_2^t)'$ . The second coordinate of the latter is then replaced during the second step by  $\beta_1^{t+1}$ . The last step finishes the movement from  $\beta_t$  to  $\beta_{t+1}$  by replacing the third coordinate  $\beta_2^t$  by  $\beta_2^{t+1}$ .

During the 1000 first iterations, we use to simulate  $\beta_i^{t+1}$ , for  $i = 0, 1, 2$ , a proposal normal distribution, centered at the current value  $\beta_i^t$ , and with variance  $C_0$  whose value is set to 0.01. In the remaining iterations, the proposal variance  $C_0$  is replaced by another whose value changes at each step using eq.(2). In this equation the choice of the parameter  $s$  can largely affect the performance of the SCAM algorithm. So, we take  $s = 2.4$ , corresponding to the optimum value presented in (Haario, Saksman & Tamminen, 2005). Concerning the parameter  $\epsilon$ , which should be included between 0 and 1, we set its value to 0.001.

#### 5.2.3 Parameterizing of the DRAM Algorithm

In order to combine MHDR algorithm with AM method, several strategies had been proposed and experimented in the literature. Among them, we retain here the one recommended by (Haario, Laine, Mira & Saksman, 2006). Its principle

is to simulate a first part of the chain using the standard MHDR algorithm with a multi-normal proposal distribution. For the rest of iterations, the proposal covariance matrix of the first step is updated, at each step, using AM method.

To simulate from the target density  $\pi(\beta|D)$ , we employ the MHDR algorithm to generate the first 1000 observations. On the  $(t + 1)^{th}$  iteration of the algorithm, the first step is to suggest a value  $y_1$  from the candidate density  $q_1 = N_3(\beta_t, C_0)$  where  $\beta_t$  is the current state of the chain and  $C_0$  denotes the covariance matrix such that  $C_0 = \text{diag}[1, 0.001, 0.0001]$ . In the second step of the algorithm, a new value  $y_2$  is generated from the candidate distribution  $q_2 = N_3(\beta_t, M_0)$ . As advised by (Green & Mira, 2001), we set  $M_0 = 0.5C_0$  so that, in stationarity, the acceptance rate would be optimum.

In the remaining iterations, we use, in the first step of the MHDR algorithm, the AM method, and this in order to adjust the proposal covariance matrix. Thus, on the  $(t + 1)^{th}$  iteration, a first value  $y_1$  is generated from the proposal distribution  $q_1 = N_3(\beta_t, C_{t+1})$  where  $C_{t+1}$  is the covariance matrix obtained by

$$C_{t+1} = \frac{5.760}{3} \text{cov}(\beta_0, \dots, \beta_t) + \frac{0.006}{3} I_3.$$

Recall that this proposal distribution is the same used to simulate from  $\pi(\beta|D)$  via the AM algorithm. During the second step of the sampling scheme, a second value  $y_2$  is generated from the candidate density  $q_2 = N_3(\beta_t, M_{t+1})$  where  $M_{t+1}$  denotes the covariance matrix such that  $M_{t+1} = 0.5C_{t+1}$ .

As indicated in section 4, the acceptance probability of the proposition  $y_1$  is defined by

$$\alpha_1(\beta_t, y_1) = \min \left[ 1, \frac{\pi(y_1)}{\pi(\beta_t)} \right].$$

In case of rejection, the second proposed candidate  $y_2$  becomes the new state of the chain with the probability

$$\alpha_2(\beta_t, y_2) = \min \left[ 1, \frac{\pi(y_2)q_1(y_2, y_1)[1 - \alpha_1(y_2, y_1)]}{\pi(\beta_t)q_1(\beta_t, y_1)[1 - \alpha_1(\beta_t, y_1)]} \right].$$

### 5.3 Experimental Results and Interpretation

To simulate from the posterior distribution  $\pi(\beta|D)$ , we run 110000 iterations using AM, SCAM and DRAM algorithms with an initial common value  $\beta_0$ , the null column vector of order three. After an initial burn-in period of 10000 iterations, we consider that each generated chain has reached its stationary distribution. Then the quantities  $\pi(A_1)$  and  $\pi(A_2)$  are solely estimated on the basis of the remaining 100000 iterations. These estimates have been repeated 100 times in order to know their variability. The results given in table-1 present the mean values of estimates and their standard errors. Whereas, those in table-2 concern the acceptance rates calculated in (Baffoun, Hajlaoui & Farhat, 2017) and in this work.

Table 1. Estimation results obtained by M-H, AM, SCAM and DRAM methods.

Algorithm	$\hat{\pi}^f(A_1)$		$\hat{\pi}^f(A_2)$	
	Average (%)	Std error ( $\times 10^{-3}$ )	Average (%)	Std error ( $\times 10^{-3}$ )
M-H <sup>(*)</sup>	95.62	2.523	4.38	2.523
AM	95.51	2.179	4.49	2.179
SCAM	95.67	2.186	4.33	2.186
DRAM	95.65	1.953	4.35	1.953

<sup>(\*)</sup> The M-H statistics are derived from (Liang & Liu, 2005).

Table 2. Acceptance rates of MCMC algorithms.

Algorithm	M-H	AM	SCAM	DRAM
Acceptance rate	0.168	0.248	0.233	0.379

Examining table-1 shows that all the tested sampling schemes lead practically to the same estimated quantities  $\pi(A_1)$  and  $\pi(A_2)$ . But in terms of precision, we can note some disparities in the obtained results. According to this criterion, DRAM algorithm seems to be the most attractive method while, the M-H algorithm appears to have the worst performances. In what follows, we will try to analyze and interpret these results function of the behavior of each considered algorithm, particularly, using the acceptance rate whose values are provided in table-2.

It is important to remark that the AM and SCAM algorithms provide very close results for the precision of the considered estimators. This finding is perfectly coherent with the literature that indicates for small dimension, the performances of

AM sampling scheme are identical to those of SCAM method. This similarity appears even in the values of the computed acceptance rate for each algorithm.

Compared with M-H algorithm, AM and SCAM methods seem to offer a tangible improvement since the standard error is decreased about 15%. Moreover, we note that these two sampling schemes tend to reject fewer propositions than MH algorithm, thus resulting in relatively high acceptance rates, quite close to the optimum value of 0.25. This shows clearly that the tuning of the proposal distribution allowed a good exploration of the target density support and, at the same time, a decrease in the correlation between the states of a same chain. It is precisely this last point which explains the observed reduction of the standard error by referring to Peskun's asymptotic variance ordering (Peskun, 1973).

Considering the performances of DRAM algorithm, we can notice that the obtained results are clearly better than that of AM method. In fact, the standard error has been decreased at about 10% while the acceptance rate is increased at about 35%. This constatation indicates that the proposed jumps, during the second step of the algorithm, are very small and then frequently accepted. Intuitively, this augmentation of the number of accepted candidates would decrease the correlation between the states of the simulated chain. This fact justifies the observed reduction of the standard error under the Peskun ordering.

Compared to the M-H method, the DRAM algorithm gives appreciably better results since the standard error is decreased by more than 20%. Moreover, the low acceptance rate realized by the M-H scheme indicates that most of the proposed values are localized in low density zones of  $\pi(\beta|D)$ . This leaves to suppose that the used proposal distribution is significantly different from the target density. The DRAM algorithm allowed overcoming this failure by adjusting the covariance matrix of the instrumental distribution so as to correspond with that of the target density. In addition, this algorithm has the advantage to propose more than one value per iteration, that ensures a better exploration of the support of the distribution of interest. Thus, in view of its exceptional proprieties, the DRAM scheme constitutes an interesting alternative to the M-H algorithm.

## 6. Conclusion

In this paper, three advanced MCMC methods: AM, SCAM and DRAM algorithms are studied experimentally in the context of estimating non-standard discrete distributions. The performance criterion used is the precision of the frequency estimator. The obtained results reflect the superiority of the DRAM algorithm over to the other considered sampling schemes. We also deduced that the performance of AM and SCAM algorithms are almost similar and significantly superior to that of the M-H sampler, whose results are presented in (Liang & Liu, 2005). All these results prove the high performance of adaptive MCMC methods, and more particularly the DRAM sampler. However, this powerful algorithm has the disadvantage to be expensive in terms of computing time.

## References

- Ashford, J. R. (1959). An approach to the analysis of data for semi-quantal responses in biological assay. *Biometrics*, 15, 573-581.
- Baffoun, H., Hajlaoui, M., & Farhat, A. (2017). Equation-solving estimator based on Metropolis-Hastings algorithm with delayed rejection. *Communications in Statistics: Simulation and Computation*, 46 (4), 3103-3111.
- Bardenet, R., Cappé, O., Fort, G., & Kégl, B. (2015). Adaptive MCMC with online relabeling. *Bernoulli*, 21 (3), 1304-1340. <https://doi.org/10.3150/13-BEJ578>
- Green, P. J. & Mira, A. (2001). Delayed rejection in reversible jump Metropolis-Hastings. *Biometrika*, 88, 1035-1053. <https://doi.org/10.1093/biomet/88.4.1035>
- Haario, H., Laine, M., Mira, A., & Saksman, E. (2006). DRAM: Efficient adaptive MCMC. *Statistics and Computing*, 16 (4), 339-354. <https://doi.org/10.1007/s11222-006-9438-0>
- Haario, H., Saksman, E., & Tamminen, J. (2001). An adaptive Metropolis algorithm. *Bernoulli*, 7, 223-242. <https://doi.org/10.2307/3318737>
- Haario, H., Saksman, E., & Tamminen, J. (2005). Componentwise adaptation for high dimensional MCMC. *Computational Statistics*, 20 (2), 265-273.
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57, 97-109.
- Liang, F., & Liu, C. (2005). Efficient MCMC estimation of discrete distributions. *Computational Statistics and Data Analysis*, 49, 1039-1052.
- Liang, F., Liu, C., & Carroll, R. J. (2010). Advanced Markov chain Monte Carlo Methods: Learning from Past Samples.

*John Wiley & Sons*, Ltd, Chichester, UK.

Mira, A. (1998). Ordering, Slicing and Splitting Monte Carlo Markov Chains. *PhD thesis*, University of Minnesota.

Peskun, P. H. (1973). Optimum Monte Carlo sampling using Markov chains. *Biometrika*, 60, 607-612.  
<https://doi.org/10.1093/biomet/60.3.607>

Saksman, E., & Vihola, M. (2010). On the ergodicity of the adaptive Metropolis algorithm on unbounded domains. *Annals of Applied Probability*, 20, (6), 2178-2203.

Vihola, M. (2012). Robust adaptive Metropolis algorithm with coerced acceptance rate. *Statistics and Computing*, 22, (5), 997-1008.

### **Copyrights**

Copyright for this article is retained by the author(s), with first publication rights granted to the journal.

This is an open-access article distributed under the terms and conditions of the Creative Commons Attribution license (<http://creativecommons.org/licenses/by/4.0/>).