

The application of Monte Carlo methods for learning generalized linear model

Abstract

Monte Carlo method is a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. They are often used in physical and mathematical problems and are most useful when it is difficult or impossible to use other mathematical methods. Basically, many statisticians have been increasingly drawn to Monte Carlo method in three distinct problem classes: optimization, numerical integration, and generating draws from a probability distribution. In this paper, we will introduce the Monte Carlo method for calculating regression coefficients in Generalized Linear Model (GLM), especially for Logistic Regression. Our main methods are Metropolis Hastings (MH) Algorithms and Stochastic Approximation in Monte Carlo Computation (SAMC). For comparison, we also get results automatically using MLE method in R software. Then we apply some Monte Carlo algorithms to a real example study and compare the efficiency of each method.

Keywords: Monte Carlo method, GLM, MH-algorithms, SAMC

Volume 7 Issue 5 - 2018

Bochao Jia

Eli Lilly and Company, Lilly Corporate Center, USA

Correspondence: Bochao Jia, Eli Lilly and Company, Lilly Corporate Center, Indianapolis, IN 46285, USA, Email jbc409@gmail.com

Received: September 13, 2018 | **Published:** September 24, 2018

Introduction

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for sampling from a probability distribution based on constructing a Markov chain that has the desired distribution as its equilibrium distribution. The Metropolis-Hastings (MH) algorithm is one of the best known of these methods, which was developed by Metropolis et al.¹ and subsequently generalized by Hastings.² Firstly, MH algorithm has been used in physics and was little known to statisticians until Muller³ and Tierney⁴ provided important applications using this algorithm. Then Chib & Greenberg⁵ gave some recent examples including relevant Markov chain theories which made more applications appeared in the recently literature. In the next section, we will provide a brief introduction to the MH-Algorithms and its work principles. We will also discuss the three different MH-Algorithms based on different sampling methods.

As for Stochastic Approximation in Monte Carlo Computation (SAMC), it was first introduced by Liang et al.⁶ As we know, the Metropolis-Hastings (MH) algorithm and the Gibbs sampler Geman⁷ are prone to get trapped into local energy minima in simulations from a system for which the energy landscape is rugged. In terms of physics, the negative of the logarithmic density/mass function is called the energy function of the system. To overcome this problem, many advanced Monte Carlo algorithms have been proposed, such as parallel tempering Geyer,⁸ Hukushima & Nemoto,⁹ simulated tempering Marinari & Parisi,¹⁰ Geyer & Thompson,¹¹ evolutionary Monte Carlo Liang & Wong,¹² dynamic weighting Wong & Liang,¹³ multicanonical sampling Berg & Neuhaus,¹⁴ 1/k-ensemble sampling Hesselbo & Stinchcombe,¹⁵ the Wang-Landau algorithm Wang & Landau,¹⁶ equi-energy sampler Mitsutake et al.¹⁷ stochastic approximation Monte Carlo Liang et al.⁹ among others. In this paper, we adopt the SAMC algorithm to our problem, which has been proofed to be a sophisticated algorithm in both theory and applications. The basic idea of SAMC stems from the Wang-Landau algorithm and extends it to the continuum systems and it is designed to simulate data with a complex model structure.

The rest of the paper is organized as follows. In section 2, we will introduce three different sampling methods based on Metropolis

Hastings Algorithms and SAMC algorithm for parameter estimations in the logistic regression. In section 3, we will give a simple simulation on the all of the methods and compare the results with the MLE method provided in R package. In section 4, we apply the proposed method to a Adult Intelligence data. In section 5, we conclude the paper with a brief discussion.

Method

Metropolis hastings algorithms

The working principle of Markov Chain Monte Carlo methods is quite straight forward to describe. Given a target density f , we build a Markov kernel K with stationary distribution f and then generate a Markov chain $(X^{(t)})$ using this kernel so that the limiting distribution of $(X^{(t)})$ is f and integrals can be approximated according to the Ergodic Theorem. The difficulty should thus be in constructing a kernel K that is associated with an arbitrary density f . But, quite miraculously, there exist methods for deriving such kernels that are universal in that they are theoretically valid for any density f .

The Metropolis Hastings algorithm is an example of those methods. Given the target density f , it is associated with a working conditional density $q(y|x)$ that, in practice, is easy to simulate. In addition, q can be almost arbitrary in that the only theoretical requirements are that the ratio $f(y)/q(y|x)$ is known up to a constant independent of x and that $q(\cdot|x)$ has enough dispersion to lead to an exploration of the entire support of $\alpha_1(\beta^{(t)}, \beta')$. For every given q , we can then construct a Metropolis Hastings kernel such that f is its stationary distribution. Here is the general step of the Metropolis Hastings algorithm.

Algorithm (Metropolis Hastings)

- i. Initialize $x^{(0)} \sim q(x)$.
- ii. Given current value of $x^{(t)}$, generate a candidate $y \sim q(y|x^{(t)})$.
- iii. Calculate acceptance probability

$$\alpha(x^{(t)}, y) = \min\left\{\frac{f(y)q(x^{(t)}|y)}{f(x^{(t)})q(y|x^{(t)})}, 1\right\}.$$

iv. Take $x^{(t+1)} = y$ with probability $\alpha(x^{(t)}, y)$ and $x^{(t+1)} = x^{(t)}$ otherwise.

v. Iterate between step (ii) and step (iv) until converge.

In this paper, we consider the logistic regression. We want to use Metropolis Hastings algorithm to fit the parameter of the model. i.e β_0 and β_1 in the following function.

$$Y_i \sim \text{Bin}(1, \pi_i), \log \frac{\pi_i}{1 - \pi_i} = \beta_0 + x_i \beta_1, i = 1, \dots, n \quad (1)$$

And the likelihood function is

$$f(y | \beta_0, \beta_1) = \prod_{i=1}^n \left(\frac{e^{\beta_0 + x_i \beta_1}}{1 + e^{\beta_0 + x_i \beta_1}} \right)^{y_i} \left(\frac{1}{1 + e^{\beta_0 + x_i \beta_1}} \right)^{1-y_i} \quad (2)$$

$$2.0\text{cm} = \exp\{n\bar{y}\beta_0 + \beta_1 \sum_{i=1}^n x_i y_i - \log(1 + e^{\beta_0 + x_i \beta_1})\} \quad (3)$$

Consider the prior distribution of β_0 and β_1 as the independent normal distribution

$$\beta_j \sim N(\mu_j, \sigma_j^2), j = 0, 1. \quad (4)$$

Then the posterior distribution is

$$f(\beta_0, \beta_1 | y) \propto f(y | \beta_0, \beta_1) \pi(\beta_0, \beta_1)$$

$$2.0\text{cm} \propto \exp\{n\bar{y}\beta_0 + \beta_1 \sum_{i=1}^n [x_i y_i - \log(1 + e^{\beta_0 + x_i \beta_1})]\}$$

$$2.0\text{cm} - \frac{(\beta_0 - \mu_0)^2}{2\sigma_0^2} - \frac{(\beta_1 - \mu_1)^2}{2\sigma_1^2}$$

And we need to get the Markov sequence of the parameters. We propose to consider three sampling methods based on the general Metropolis Hastings algorithm, which is independent sampling, dependent sampling and individual sampling

Independent sampling

Assuming that the parameters β_0 and β_1 are independent, we can generate $\beta^{(t)} = (\beta_0^{(t)}, \beta_1^{(t)})$ from the proposal distribution $N(\beta^{(t-1)}, \text{diag}(\sigma_0^2, \sigma_1^2))$. Then the independent sampling can be stated as following.

Algorithm (Independent sampling)

- Initialize $\beta^{(0)} = (\beta_0^{(0)}, \beta_1^{(0)})$.
- Generate β' from the proposal distribution $N(\beta^{(t-1)}, \text{diag}(\sigma_0^2, \sigma_1^2))$.
- Calculate acceptance probability
- $\alpha(\beta^{(t-1)}, \beta') = \min\left\{\frac{f(y | \beta_0', \beta_1') q(\beta_0', \beta_1')}{f(y | \beta_0^{(t-1)}, \beta_1^{(t-1)}) q(\beta_0^{(t-1)}, \beta_1^{(t-1)})}, 1\right\}$
- We accept $\beta^{(t)} = \beta'$ with probability $\alpha(\beta^{(t-1)}, \beta')$ or $\beta^{(t)} = \beta^{(t-1)}$ otherwise.
- Iterate between step (ii) and step (iv) until converge.

Dependent sampling

If the parameters are not independent, the convergence of stochastic sequence may be less efficient. So we consider the multinomial prior for β_0 and β_1 . In this case, we use the Fisher information matrix $H(\beta)$ to build the proposal distribution.

$$\beta' \sim q = N(\beta, c_\beta^2 [H(\beta)]^{-1}) \quad (5)$$

Where c_β is a regulation parameter, which can be adjust to reach the target acceptance rate. Based on the likelihood function, we get the information matrix

$$H(\beta) = X^T \text{diag}(h_i) X + \Sigma_\beta^{-1} \quad (6)$$

Where Σ_β is the prior covariance matrix of β , $h_i = \exp(\beta_0 + \beta_1 x_i) / (1 + \exp(\beta_0 + \beta_1 x_i))^2$, $X = (\mathbf{1}_n, x)$ is a $2 \times n$ matrix.

Then we give the algorithm for this method:

Algorithm (Dependent Sampling)

- Initialize $\beta^{(0)} = (\beta_0^{(0)}, \beta_1^{(0)})$.
- Calculate Fisher Information matrix
- $\text{diag}(h_i) = \text{diag}\left(\frac{\exp(\beta_0^{(t-1)} + \beta_1^{(t-1)} x_i)}{(1 + \exp(\beta_0^{(t-1)} + \beta_1^{(t-1)} x_i))^2}\right)$
- $H(\beta^{(t-1)}) = X^T \text{diag}(h_i) X + \Sigma_{\beta^{(t-1)}}^{-1} = c_{\beta^{(t-1)}}^2 [H(\beta^{(t-1)})]^{-1}$
- Generate β' from the proposal distribution $N(\beta^{(t-1)}, S_{\beta^{(t-1)}})$.
- Calculate acceptance probability
- $\alpha(\beta^{(t-1)}, \beta') = \min\left\{\frac{f(y | \beta_0', \beta_1') \pi(\beta_0', \beta_1') q(\beta_0', \beta_1' | \beta', S_{\beta'})}{f(y | \beta_0^{(t-1)}, \beta_1^{(t-1)}) \pi(\beta_0^{(t-1)}, \beta_1^{(t-1)}) q(\beta_0^{(t-1)}, \beta_1^{(t-1)} | \beta^{(t-1)}, S_{\beta^{(t-1)}})}, 1\right\}$
- We accept $\beta^{(t)} = \beta'$ with probability $\alpha(\beta^{(t-1)}, \beta')$ or $\beta^{(t)} = \beta^{(t-1)}$ otherwise.
- Iterate between step (ii) and step (v) until converge.

Individual sampling

In the Dependent Sampling method, we need to adjust the regulation parameter c_β that make to reach the target acceptance rate. In order to avoid this process, we propose another MH sampling algorithm, i.e. Individual Sampling. For each parameter, we generate it individually from the normal proposal distribution. We only give the algorithm in two parameters condition, since when the dimension is too large and we need to calculate acceptance rate for each parameter, which is too time-consuming.

Algorithm (Individual Sampling)

- Initialize $\beta^{(0)} = (\beta_0^{(0)}, \beta_1^{(0)})$.
- Generate β_0' from the proposal distribution $N(\beta_0^{(t-1)}, \sigma_0^2)$.
- Set $\beta' = (\beta_0', \beta_1^{(t-1)})$ and calculate acceptance rate
- $\alpha_0(\beta^{(t-1)}, \beta') = \min\left\{\frac{f(y | \beta_0', \beta_1^{(t-1)}) \pi(\beta_0', \beta_1^{(t-1)})}{f(y | \beta_0^{(t-1)}, \beta_1^{(t-1)}) \pi(\beta_0^{(t-1)}, \beta_1^{(t-1)})}, 1\right\}$
- We accept $\beta^{(t)} = \beta'$ with probability $\alpha_0(\beta^{(t-1)}, \beta')$ or $\beta^{(t)} = \beta^{(t-1)}$ otherwise.
- Generate β_1' from the proposal distribution $N(\beta_1^{(t)}, \sigma_1^2)$.
- Set $\beta' = (\beta_0^{(t)}, \beta_1')$ and calculate acceptance rate
- $\alpha_1(\beta^{(t)}, \beta') = \min\left\{\frac{f(y | \beta_0^{(t)}, \beta_1') \pi(\beta_0^{(t)}, \beta_1')}{f(y | \beta_0^{(t)}, \beta_1^{(t)}) \pi(\beta_0^{(t)}, \beta_1^{(t)})}, 1\right\}$

- ix. We accept $\beta^{(t+1)} = \beta'$ with probability $\alpha_1(\beta^{(t)}, \beta')$ or $\beta^{(t+1)} = \beta^{(t)}$ otherwise.
- x. Iterate between step (ii) and step (vii) until converge.

Stochastic approximation Monte Carlo computation (SAMC)

Aforementioned, we introduce three MH algorithms for learning the Logistics regression. In this part, we will illustrate the Stochastic Approximation Monte Carlo Computation (SAMC) to deal with this problem.

Consider the sampling distribution that $p(x) = cp_0(x)$, where c is a constant, x is generated from the sample space \mathcal{X} . We let E_1, \dots, E_m donate m disjoint regions that from a partition of \mathcal{X} , which can be partitioned according to any function of x such as the energy function $U(x)$ as follows: $E_1 = \{x : U(x) \leq u_1\}, E_2 = \{x : u_1 < U(x) \leq u_2\}, \dots, E_m = \{x : U(x) \geq u_m\}$.

Then we let $\hat{g}_i^{(t)}$ donate the estimate of g_i and $\theta_{ti} = \log(\hat{g}_i^{(t)})$ and rewritten the invariant distribution $\hat{p}(x)$ in the generalized Wang-Landau (GWL) algorithm as

$$p_{\theta_t}(x) \propto \sum_{i=1}^m \frac{\psi(x)}{e^{\theta_{ti}}} I(x \in E_i) \quad (7)$$

For theoretical simplicity, we assume that $\theta_t \in \Theta$ for all t , where Θ is a compact set. Let $\pi = (\pi_1, \dots, \pi_m)$ be an m -vector with $0 < \pi_i < 1$ and $\sum_{i=1}^m \pi_i = 1$, which defines the desired sampling frequency for each of the sub regions. Henceforth, π is called the desired sampling distribution. Let γ_t be a positive, no decreasing sequence satisfying

$$(a) \sum_{t=1}^{\infty} \gamma_t = \infty \text{ and } (b) \sum_{t=1}^{\infty} \gamma_t < \infty \quad (8)$$

For some $\zeta \in (1, 2)$. For example, in this article we set

$$\gamma_t = \frac{t_0}{\max(t_0, t)}, t = 1, 2, \dots, \quad (9)$$

For some specified value of $t_0 > 1$.

In the logistic regression model, the invariant distribution

$$p_{\theta_t}(\beta) \propto \exp\{n\bar{y}\beta_0 + \beta_1 \sum_{i=1}^n x_i y_i - \log(1 + e^{\beta_0 + x_i \beta_1})\} \quad \text{and the}$$

proposal distribution $q(\beta^{(t)}) = N(\beta^{(t-1)}, \mathbf{I}_2)$. With the foregoing notation, one iteration of SAMC can be described as follows:

Algorithm (SAMC)

Generate a sample $\beta^{(t)} = (\beta_0^{(t)}, \beta_1^{(t)})$ by a single MH update with the target distribution given by

$$p_{\theta_t}(\beta) \propto \exp\{n\bar{y}\beta_0 + \beta_1 \sum_{i=1}^n x_i y_i - \log(1 + e^{\beta_0 + x_i \beta_1})\}$$

Generate β' according to the proposal distribution $q(\beta' | \beta^{(t)}) = N(\beta^{(t)}, \mathbf{I}_2)$. If $J(\beta') \notin S$, then update S to $S \cup J(\beta')$, where S denote the collection of indices of the sub regions from which a sample has been proposed and $J(\beta')$ denote the index of the sub region of sample β' .

Calculate the ratio

$$r = e^{\theta_{U(\beta^{(t)})} - \theta_{U(\beta')}} \frac{\psi(\beta') q(\beta^{(t)} | \beta')}{\psi(\beta^{(t)}) q(\beta' | \beta^{(t)})}$$

Accept the proposal with probability $\min(r, 1)$. If accepted, set $\beta^{(t+1)} = \beta'$, otherwise, set $\beta^{(t+1)} = \beta^{(t)}$.

For all $i \in S$, update θ_i to θ_{t+1} as follows:

$$\theta_{t+1,i} = \theta_{t,i} + \gamma_{t+1} (e_{t+1,i} - \pi_i)$$

where $e_{t+1,i} = 1$ if $\beta^{(t+1)} \in E_i$, and 0 otherwise.

Simulation study

In this part, we generate simple logistics regression data and use the Monte Carlo method to fit the model.

$$\log \frac{\pi_i}{1 - \pi_i} = \beta_0 + x_i \beta_1, i = 1, \dots, n \quad (10)$$

Based on the model and given the value of β_0 and β_1 , we generate 1000 samples of x_i .

$$x_i \sim N(1, 1) \pi_i = \frac{\exp(\beta_0 + x_i \beta_1)}{1 + \exp(\beta_0 + x_i \beta_1)}, i = 1, \dots, n \quad (11)$$

Then y_i is generated from binomial π_i distribution. We do iterations for 1000 times and calculate the mean and variance of the $(\hat{\beta}_0, \hat{\beta}_1)$ in different methods (Table 1).

Based on the Table 1, we find that both MH algorithms and SAMC have good performance on calculating the parameters in logistics regression while SAMC has smallest variance which indicates a better convergence rate and robustness. Since the independence of β_0 and β_1 , also the $\text{Cov}(\beta_0, \beta_1)$ is close to 0, all the three MH algorithms perform similarly. As for SAMC algorithm, we have to partition the sample space based on the energy function. The energy regions can set up the initial values spanned in the full model space and therefore can have faster convergence rate and lower variance than other MH algorithm. Finally, we use the MLE method, i.e the 'glm' function in R to estimate the parameters for comparison. It shows that SAMC algorithm even achieve better performance than the MLE method in terms of the variance of the estimators. Therefore, both MH and SAMC algorithms are eligible to obtain the consistent estimators in the logistic regression but even for this simple problem, SAMC still can enjoy better performance than others.

Table 1 Comparison of parameter estimations and variances for 5 pairs of (β_0, β_1)

(β_0, β_1)		(0.1,0.2)	(0.6,0.3)	(1,-3)	(2,0.4)	(-3,2)
Independent	mean	(0.10,0.17)	(0.56,0.27)	(1.22,-3.01)	(1.78,0.35)	(-3.20,1.92)
	variance	0.09,0.07	0.11,0.08	0.15,0.23	0.13,0.10	0.25,0.22
Dependent	mean	(0.10,0.18)	(0.59,0.25)	(1.12,-2.87)	(2.11,0.38)	(-3.13,2.05)
	variance	0.09,0.07	0.09,0.08	0.15,0.23	0.13,0.10	0.25,0.22

Table Continued..

(β_0, β_1)		(0.1,0.2)	(0.6,0.3)	(1,-3)	(2,0.4)	(-3,2)
Individual	mean	(0.10,0.17)	(0.57,0.26)	(1.12,-2.88)	(2.13,0.39)	(-2.98,2.08)
	variance	0.09,0.06	0.11,0.08	0.15,0.23	0.13,0.10	0.25,0.22
SAMC	mean	(0.09,0.19)	(0.62,0.27)	(1.09,-2.99)	(2.11,0.43)	(-3.19,2.01)
	variance	0.06,0.05	0.11,0.07	0.11,0.11	0.09,0.10	0.24,0.20
MLE	mean	(0.10,0.19)	(0.59,0.3)	(1.01,-2.97)	(2.01,0.40)	(-3.01,2.01)
	variance	0.09,0.07	0.21,0.16	0.19,0.27	0.24,0.22	0.16,0.24

Real example

In this part, we apply Monte Carlo methods into a real example data set. Our dataset is obtained from the Wechsler Adult Intelligence Scale (WAIS), which is a test designed to measure intelligence in adults and older adolescents and it is currently in its fourth edition (WAIS-IV). The original WAIS was published in February 1955 by David Wechsler, as a revision of the Wechsler-Bellevue Intelligence Scale that had been released in 1939. The propose of this test is to find the relationship between intelligence and Senile Dementia among older people. In our score scale, people (with and without Senile Dementia) were asked to do some tasks individually. When all tasks completed, they will get a score, ranging from 0-20. Therefore, the response y is a binary response that reflects whether people have Senile Dementia. The WAIS score is the explanation variables. Based on data, we need to build the logistic model and estimate the parameters.

Firstly, we apply the MH Independent Sampling Algorithm. We set the length of chain equals to 10000 and get the plot of parameters of β_0 , β_1 . As showed in the lower two figures of Figure 1, the two

lines does not coincide, which indicate that independent sampling algorithm does not converge. The independent sampling algorithm assumes that parameters are independent; however, it cannot be easily satisfied in the real application. We calculate the covariance of β_0 and β_1 , which is $\text{Cov}(\beta_0, \beta_1) = -0.93$. Therefore, the independent sampling method cannot be applied here. Then we use another two MH algorithms, i.e dependent sampling and individual sampling as well as SAMC algorithm. We set the length of chain equals to 5000 and plot the cumulative mean curves for each method. As showed in Figure 2, all the methods converge when iteration grows while the SAMC converges faster than other, i.e; the two curves for SAMC coincide at around iteration 2000 while others need more iteration. To access the accuracy of estimations of β_0 and β_1 , we use the results from MLE method as benchmark as compare them with the ones from the Monte Carlo methods. As showed in Table 2, the results from SAMC are more close to the MLE and also it has the smallest variance among all other methods. According to the result, β_1 is significantly less than 0, which means people who get less score have higher risk of Senile Dementia.

Table 2 Results of both value and variance for each method of (β_0, β_1)

(β_0, β_1)	Independent	Dependent	Individual	SAMC	MLE
mean	(2.68,-0.34)	(2.61,-0.34)	(2.62,-0.35)	(2.57,-0.34)	(2.40,-0.32)
variance	0.19,0.18	1.27,0.24	1.23,0.10	1.13,0.09	1.19,0.11

Discussion

In this paper, we have proposed three MH algorithms and also SAMC to estimate the parameters of logistic regression. According to the simulation study, when the parameters are uncorrelated, each method has good performance. When the parameters are highly correlated, Independent Sampling method is less efficient and even cannot converge. The results showed that SAMC algorithms can be used in both scenarios and always outperform others. Compared with other MH algorithm, SAMC need less iterations to converge and have smaller variance of estimators even compared with MLE method.

One interesting further work is to apply the Monte Carlo simulations to high-dimensional big data problem. The challenge comes from two aspects. First, the data contains complex structure and therefore, the acceptance rate for most MCMC algorithm can be slow and therefore can be less efficient. Although SAMC has been proved to be a good method for the complex data and can overcome some local trap problem, it still needs to improve to the high-dimensional case where

the likelihood function may not be intractable. Secondly, the big data have catastrophically large volumes and most single chain Monte Carlo simulations cannot apply. At each iteration, it should go through all the data which need large memory space for the computer machine and long time to generate the data at each iteration. For the big volume of data, we should consider parallel computing strategies, i.e, generate multiple Monte Carlo chains in parallel and then integrate chains to get the single estimator (Liang and Wu, 2013). Also, the high speed graphics processing unit (GPU) can be helpful for accelerating the speed of many MCMC algorithms.

In conclusion, we introduced several Monte Carlo simulation methods to estimate parameters of generalized linear model, i.e logistic regression. This problem can also be achieved by the MLE method which might be easier to calculate. However, our Monte Carlo algorithm can obtain a chain of estimations instead of a single value and therefore can provide more information about the process of reaching the true values of parameters and can be useful for further analysis.

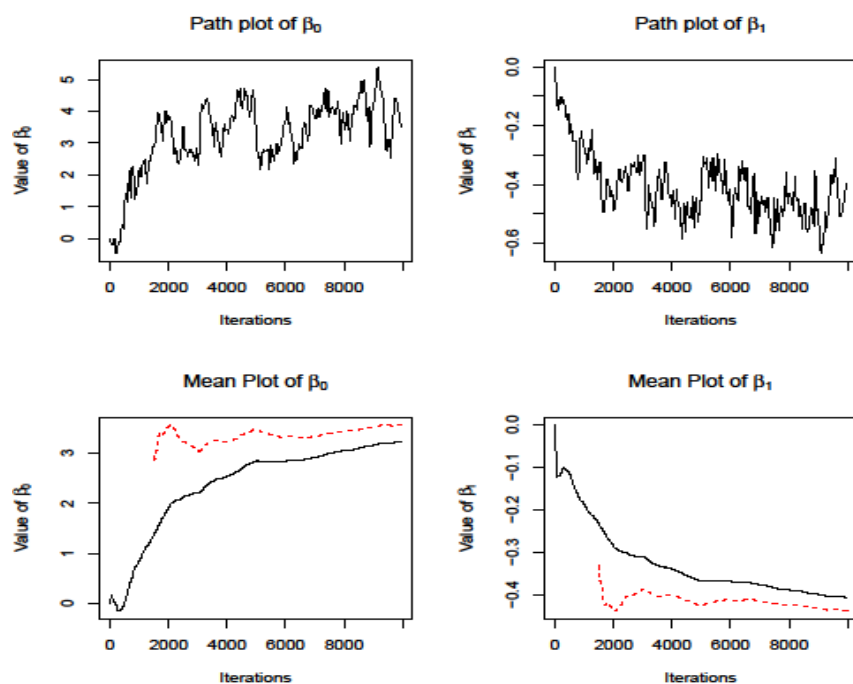


Figure 1 The path and mean plots of β_0 , β_1 for MH Independent Sampling algorithm. The upper two plots are the trace plot of estimators of β_0 and β_1 at each iteration. The lower two plots are the cumulative mean of β_0 and β_1 , respectively. The solid line calculates the cumulative mean starting at iteration 1, while the dash lines starts at step 1500.

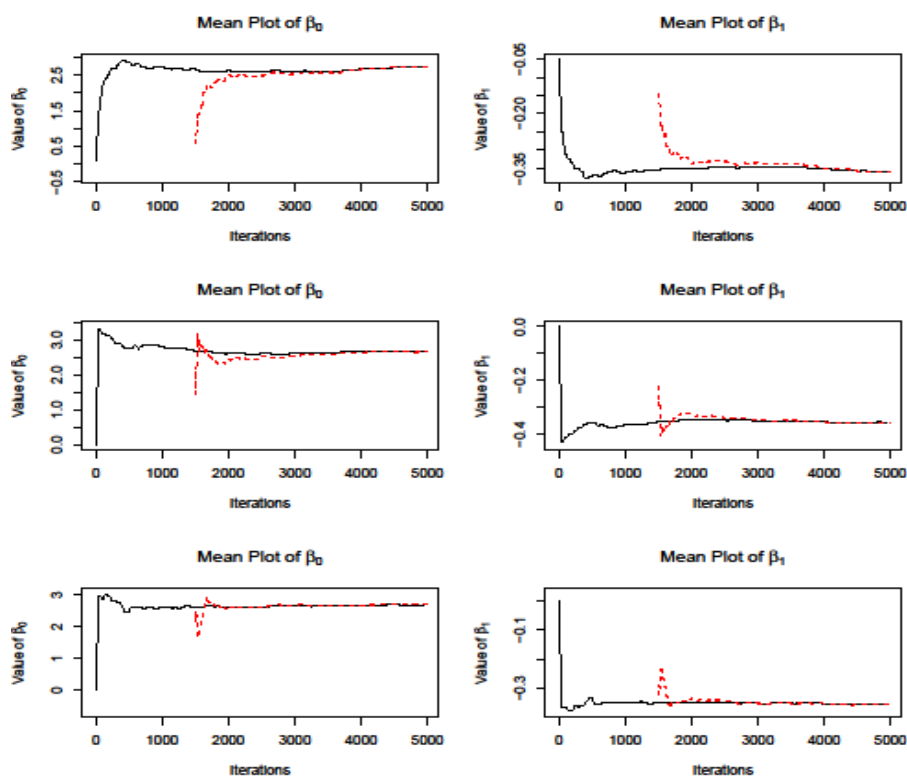


Figure 2 Comparison the cumulative mean plot of β_0 and β_1 for multiple Monte Carlo algorithms. The upper two panels denote the MH Dependent Sampling method; the middle two panels denote the MH individual sampling method; the lower two panels denote the SAMC method. The solid line calculates the cumulative mean starting at iteration 1, while the dash lines starts at step 1500.

Acknowledgement

The authors would like to thank the referee and the Editor for careful reading and comments which greatly improved the paper.

Conflict of interest

Author declares that there is no conflict of interest.

References

1. Metropolis N, Rosenbluth AW, Rosenbluth MN, et al. Equation of state calculations by fast computing machines. *The journal of chemical physics*. 1953;21:1087–1092.
2. Hastings WK. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*. 1970;57(1):97–109.
3. Müller P. *A generic approach to posterior integration and Gibbs sampling*. USA: Purdue University, Department of Statistics. 1991.
4. Tierney L. Markov chains for exploring posterior distributions. *The Annals of Statistics*. 1994;22(4):1701–1728.
5. Chib S, Greenberg E. Markov chain Monte Carlo simulation methods in econometrics. *Econometric theory*. 1996;12:409–431.
6. Liang Liu, Carroll, Carroll RJ. Stochastic approximation in Monte Carlo computation. *Journal of the American Statistical Association*. 2007;102:305–320.
7. Geman S, Geman D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*. 1994;721–741.
8. Geyer CJ. Markov chain Monte Carlo maximum likelihood. 1991.
9. Hukushima K, Nemoto K. Exchange Monte Carlo method and application to spin glass simulations. *Journal of the Physical Society of Japan*. 1996;65:1604–1608.
10. Marinari E, Parisi G. Simulated tempering: a new Monte Carlo scheme. *Europhysics Letters*. 1992;19:451.
11. Geyer CJ, Thompson EA. Annealing Markov chain Monte Carlo with applications to ancestral inference. *Journal of the American Statistical Association*. 1995;90:909–920.
12. Liang F, Wong WH. Evolutionary Monte Carlo for protein folding simulations. *The Journal of Chemical Physics*. 2001;115:3374–3380.
13. Wong WH, Liang F. Dynamic weighting in Monte Carlo and optimization. *Proceedings of the National Academy of Sciences*. 1997;94:14220–14224.
14. Berg BA, Neuhaus T. Multicanonical ensemble: A new approach to simulate first-order phase transitions. *Physical Review Letters*. 1992;68:1–9.
15. Hesselbo B, Stinchcombe RB. Monte Carlo simulation and global optimization without parameters. *Physical review letters*. 1995;74:2151.
16. Wang F, Landau D. Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical review letters*. 2001;86:2050.
17. Mitsutake A, Sugita Y, Okamoto Y. Replica-exchange multicanonical and multicanonical replica-exchange Monte Carlo simulations of peptides. I. Formulation and benchmark test. *The Journal of chemical physics*. 2003;118:6664–6675.