Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9248318, 11 pages
https://doi.org/10.1155/2018/9248318

Research Article

Multiple-Try Simulated Annealing Algorithm for Global Optimization

Wei Shao [1] **and Guangbao Guo** [2]

[1] *School of Management, Qufu Normal University, Rizhao, Shandong 276826, China*
[2] *Department of Statistics, Shandong University of Technology, Zibo 255000, China*

Correspondence should be addressed to Wei Shao; wshao1031@gmail.com

Received 18 March 2018; Revised 23 May 2018; Accepted 29 May 2018; Published 17 July 2018

Academic Editor: Guillermo Cabrera-Guerrero

Simulated annealing is a widely used algorithm for the computation of global optimization problems in computational chemistry and industrial engineering. However, global optimum values cannot always be reached by simulated annealing without a logarithmic cooling schedule. In this study, we propose a new stochastic optimization algorithm, i.e., simulated annealing based on the multiple-try Metropolis method, which combines simulated annealing and the multiple-try Metropolis algorithm. The proposed algorithm functions with a rapidly decreasing schedule, while guaranteeing global optimum values. Simulated and real data experiments including a mixture normal model and nonlinear Bayesian model indicate that the proposed algorithm can significantly outperform other approximated algorithms, including simulated annealing and the quasi-Newton method.

1. Introduction

Since the 21st century, the modern computers have greatly expanded the scientific horizon by facilitating the studies on complicated systems, such as computer engineering, stochastic process, and modern bioinformatics. A large volume of high dimensional data can easily be obtained, but their efficient computation and analysis present a significant challenge.

With the development of modern computers, Markov chain Monte Carlo (MCMC) methods have enjoyed a enormous upsurge in interest over the last few years [1, 2]. During the past two decades, various advanced MCMC methods have been developed to successfully compute different types of problems (e.g., Bayesian analysis, high dimensional integral, and combinational optimization). As an extension of MCMC methods, the simulated annealing (SA) algorithm [1–3] has become increasingly popular since it was first introduced by Kirkpatrick et al. (1983). As Monte Carlo methods are not sensitive to the dimension of data sets, the SA algorithm plays an important role in molecular physics, computational chemistry, and computer science. It has also been successfully applied to many complex optimization problems.

Several improved optimization methods have been proposed recently [4, 5] and successfully applied to polynomial and vector optimization problems, min–max models, and so on [6–10]. Although Sun and Wang (2013) discussed the error bound for generalized linear complementarity problems, all of these improved methods were designed for special optimization problems and not for global optimization problems. The SA algorithm is a global optimization algorithm that can obtain global optimization results with slowly decreasing temperature schedule. However, Holley et al. (1989) pointed out that only with the use of a "logarithmic" cooling schedule could the SA algorithm converge to the global minimum with probability one [11, 12]. Liang et al. (2014) improved the SA algorithm by introducing the simulated stochastic approximation annealing (SAA) algorithm [13, 14]. Such algorithm can work with a square-root cooling schedule in which the temperature can decrease much faster than that in a "logarithmic" cooling schedule. Karagiannis et al. (2017) extended the SAA algorithm by using population Monte Carlo ideas and introduced the parallel and interacting stochastic approximation annealing (PISAA) algorithm [15].

In the present study, we propose a variation of the SA algorithm, i.e., the multiple-try Metropolis based simulated

annealing (MTMSA) algorithm, for global optimization. The MTMSA algorithm is a combination of the SA algorithm and multiple-try Metropolis (MTM) algorithm [16]. The MTM algorithm, which allows several proposals from different proposal distributions in the multiple-try step simultaneously, achieves a higher rate of convergence than the standard Metropolis algorithm (e.g., random walk Metropolis algorithm) [1, 2, 17]. Thus, the MTMSA algorithm can guarantee that global minima are reached with a rapidly decreasing cooling schedule. Comparing with PISAA, which should run on multicore computer to their advantage, the MTMSA often owns high convergent rate by use of the efficient vector operation with MATLAB or R on personal and super computer. Simulation and real data examples show that, under the framework of the multiple-try algorithm, the MTMSA algorithm can reach global minima under a rapidly decreasing cooling schedule relative to that of the SA algorithm.

The remainder of this paper is organized as follows. Section 2 describes the framework of the MTMSA algorithm. Section 3 illustrates the comparison of the MTMSA algorithm with other optimization methods in a mixture normal model. Section 4 presents the application of the MTMSA algorithm to Bayesian analysis and half-space depth computation through real data sets. Finally, Section 5 summarizes the conclusions derived from the study.

2. MTMSA Algorithm

2.1. Overview of SA Algorithm. The SA algorithm originates from the annealing process, which is a thermodynamic process used to attain a low energy state in condensed matter physics [1]. The process comprises two steps. The first state is the high temperature state, in which solids transform into liquid and particles move freely to ward ideal. Then the temperature drops to zero slowly and the movement of the particles become restricted such that the desired structure is achieved. Realizing that the Metropolis algorithm [18] can be used to simulate the movements of particles, Kirkpatrick et al. (1983) proposed a computer simulation based physical annealing process, i.e., the SA algorithm.

Suppose our goal is to find the minimum value of $h(\mathbf{x})$, $\mathbf{x} \in D$. This goal is equivalent to the search for the maximum value of $\exp\{-h(\mathbf{x})/T\}$, $\mathbf{x} \in D$, with any positive temperature T. Let $T_1 > T_2 > \cdots > T_k > \cdots$ be a decreasing temperature sequence, with large T_1 and $\lim_{k\longrightarrow+\infty} T_k = 0$. In every temperature T_k, we use the Metropolis-Hastings (MH) algorithm (or Gibbs sampling algorithm [19]) to update the Markov chain N_k times, with $\pi_k(\mathbf{x}) \propto \exp\{-h(\mathbf{x})/T_k\}$ as its stationary distribution. When k is increasing, an increasing number of samples concentrate in the maximum value nearby. The SA algorithm can be summarized as follows:

(1) Initialize $\mathbf{x}^{(\mathbf{0})}$ with starting temperature T_1.

(2) At current temperature T_k, update the Markov chain N_k times, with $\pi_k(\mathbf{x})$ as its stationary distribution, and transmit the last state $\mathbf{x}$ to the next temperature.

(3) Update k to $k + 1$.

The SA algorithm can reach the global optimum when the temperature sequence decreases slowly (e.g., the inverse logarithmic rate, i.e., the order of $O(\log(L_k)^{-1})$, where $L_k = N_1 + \cdots + N_k$ [1, 2, 11, 12]). However, no one can afford to use such a slow cooling schedule. Various improved SA algorithms have thus been designed and to overcome the excessively slow cooling schedule and to successfully resolve various optimization problems in industries and commerce [14, 20–23]. Realizing that the MTM algorithm can overcome the "local-trap" problem and enjoy a high convergence rate, we propose the MTMSA algorithm, which is a combination of the MTM algorithm and the SA algorithm.

2.2. MTMSA Algorithm. The Metropolis algorithm is the first iterative sampling algorithm of the MCMC algorithm. Hastings extended this algorithm by allowing the proposal distribution to be an asymmetric distribution, i.e., the MH algorithm [24].

The MH algorithm is an iterative MCMC sampling algorithm, whose iterative points $\{x_0, x_1, \ldots, x_n, \ldots\}$ show a limiting distribution $\pi(x)$. The challenge of using the MH algorithm is that it tends to suffer from the "local-trap" problem when the target distribution function $\pi(x)$ is a multimodal distribution [2, 19]. It eventually impedes the convergence of the SA algorithm. The MTM algorithm can overlap the "local-trap" problem. The MTM algorithm allows several proposal points simultaneously and selects the best one as the next sampling point while keeping the stationary distribution unchanged. Thus, combining the MTM algorithm and SA algorithm yields the MTMSA algorithm.

Suppose the global optimization problem is

$$\min_{\mathbf{x}\in D} \quad h(\mathbf{x}). \tag{1}$$

The whole procedure of the MTMSA algorithm for problem (1) is summarized below.

(1) Set the temperature parameters T_{max}, T_{min}, $a \in (0, 1)$, the length of the Markov chain N_k, and the number of multiple-try m in the MTM algorithm. Initialize the state of the Markov chain $\mathbf{x}$, and set $k = 1$.

(2) At current temperature $T_k = T_{max} \times a^k$, let

$$\pi_k(\mathbf{x}) \propto \exp\left\{-\frac{h(\mathbf{x})}{T_k}\right\} \tag{2}$$

be the stationary distribution. For $l = 1, 2, \ldots, N_k$, use the MTM algorithm to update the Markov chain N_k times.

(2.1) Propose a "proposal set" of size m from $T(\mathbf{x}, \cdot)$, denoted as $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}\}$, where $\mathbf{x_i} \in D$ and $T(\mathbf{x}, \cdot)$ is any symmetric proposal transform distribution.

(2.2) Randomly choose a proposal state $\mathbf{x}^*$ from $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}\}$ with probability $\{\pi_k(\mathbf{x_1}), \pi_k(\mathbf{x_2}), \ldots, \pi_k(\mathbf{x_m})\}$.

(2.3) Propose a "reference set" of size m, denoted as $\{\tilde{\mathbf{x}}_\mathbf{1}, \tilde{\mathbf{x}}_\mathbf{2}, \ldots, \tilde{\mathbf{x}}_\mathbf{m}\}$, where $\{\tilde{\mathbf{x}}_\mathbf{1}, \ldots, \tilde{\mathbf{x}}_\mathbf{m-1}\}$ is proposed from $T(\mathbf{x}^*, \cdot)$, and set $\tilde{\mathbf{x}}_\mathbf{m} = \mathbf{x}$.

Input: $T_{max}, T_{min}, a, N_k, m, h(\mathbf{x}), T(\mathbf{x}, \mathbf{y})$
Output: $\mathbf{x}$
(1) **Initialize:** $\mathbf{x} = \mathbf{1}, k = 1, T_1 = T_{max}$
(2) **while** $T_k > T_{min}$ **do**
(3) set $\pi_k(\mathbf{x}) \propto \exp\{-h(\mathbf{x})/T_k\}$;
(4) **for** $l = 1$ to N_k **do**
(5) $s_p = 0$;
(6) $s_r = 0$;
(7) **for** $i = 1$ to m **do**
(8) sample $\mathbf{x_i}$ from $T(\mathbf{x}, \cdot)$;
(9) $p_i = \pi_k(\mathbf{x_i})$;
(10) $s_p = s_p + p_i$;
(11) choose $\mathbf{x}^*$ from $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}\}$ with probability $\{p_1, p_2, \ldots, p_m\}$;
(12) **for** $j = 1$ to $m - 1$ **do**
(13) sample $\tilde{\mathbf{x}}_\mathbf{j}$ from $T(\mathbf{x}^*, \cdot)$;
(14) $s_r = s_r + \pi_k(\tilde{\mathbf{x}}_\mathbf{j})$;
(15) $s_r = s_r + \pi_k(\mathbf{x})$;
(16) set $r = \min\{s_p/s_r, 1\}$;
(17) sample u from uniform distribution in $[0, 1]$;
(18) **if** $u < r$ **then**
(19) set $\mathbf{x} = \mathbf{x}^*$;
(20) $k = k + 1$;
(21) $T_k = T_{max} \times a^k$;
(22) return $\mathbf{x}$;

Algorithm 1: MTMSA algorithm used to detect the minimum of $h(\mathbf{x})$, $\mathbf{x} \in D$.

(2.4) Calculate the generalized Metropolis ratio.

$$r = \min\left\{\frac{\pi_k(\mathbf{x_1}) + \pi_k(\mathbf{x_2}) + \cdots + \pi_k(\mathbf{x_m})}{\pi_k(\tilde{\mathbf{x}}_\mathbf{1}) + \pi_k(\tilde{\mathbf{x}}_\mathbf{2}) + \cdots + \pi_k(\tilde{\mathbf{x}}_\mathbf{m})}, 1\right\}. \quad (3)$$

Then update the current state of the Markov chain with probability r. Set $\mathbf{x} = \mathbf{x}^*$; otherwise, reject it, and keep $\mathbf{x}$ unchanged.

(3) If $T_k < T_{min}$, output the last solution $\mathbf{x}$ and the minimum value of (1) of the whole procedure; otherwise, update k to $k + 1$, and proceed to step (2).

Furthermore, **Algorithm** 1 gives the pseudocode of MTMSA algorithm for the computation of problem (1).

The convergence of the MTMSA algorithm can be obtained from the stationary distribution of the MTM algorithm (i.e., the detailed balance condition of the MTM algorithm [1]). Theoretically, when T_k approaches zero and the step number of the MTM algorithm is sufficiently large, all samples drawn from π_k would be in the vicinity of the global minimum of $h(\mathbf{x})$ in D.

The next proposition gives the computation complex of the MTMSA algorithm.

Proposition 1. *The computation complex of the MTMSA algorithm is*

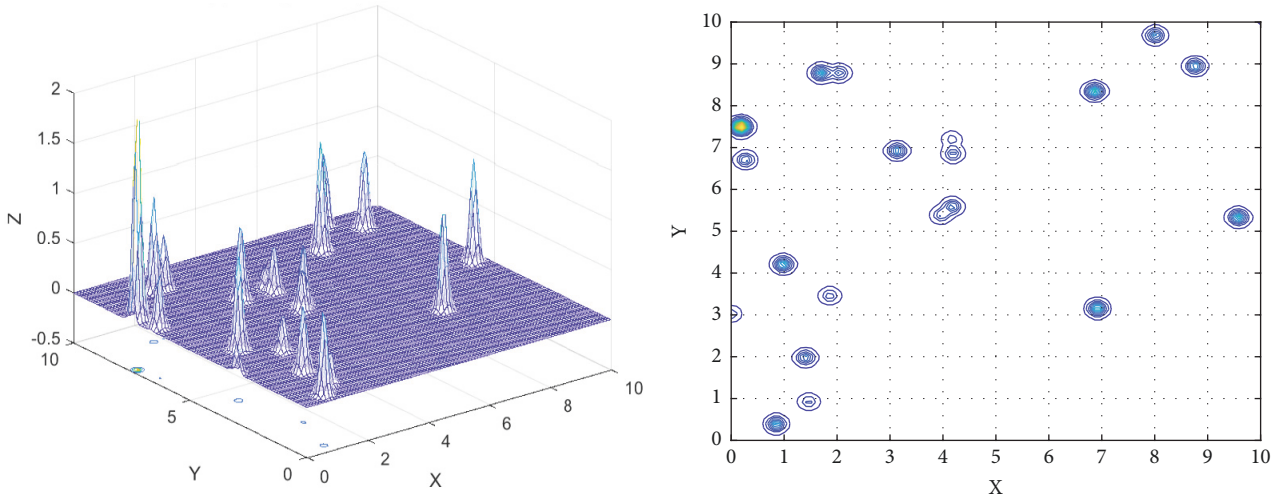
$$O(Nnm), \quad (4)$$

where N is the length of decreasing cooling temperature, n is the frequency of the Markov chain update, and m is the number of multiple-try points.

Proof. The proof of the proposition directly follows the procedure of the MTMSA algorithm described above. The decreasing cooling temperature and the length of the Markov chain are the external loops of the MTMSA algorithm. By combining the external loops with the internal loop of the multiple-try model, we then complete the proof of the proposition. □

The proposition indicates that the computation complex of the MTMSA algorithm is a polynomial in N and m. Given the computation complex of a stationary distribution $\pi_k(\mathbf{x})$, the computation complex of the MTMSA algorithm is not greater than the polynomial in d (where d is the dimension of $\mathbf{x}$).

The MTMSA algorithm has many advantages over other approximation algorithms. Compared with traditional optimization algorithms (such as the Nelder-Mead (NM) method [25] and the quasi-Newton (QN) method [26]), which are local optimization methods, the MTMSA algorithm gets more accurate results often, as shown in our simulated multimodal experiment. In practice, the speed of the MTMSA algorithm is generally high, particularly for an efficient vector operation (or parallel computing) with MATLAB or R in the evaluation of multiple-try points. The MTMSA algorithm clearly outperforms the SA algorithm in our experiment. Furthermore, by setting the number of multiple-try points $m = 1$, we can obtain a special case of the MTMSA algorithm, that is, the SA algorithm. Simulated and real data examples in subsequent sections show the advantage of the MTMSA over other approximated algorithms.



(a) Mesh plot of objective function

(b) Contour plot of objective function

Figure 1: Mesh and contour plot of the objective function.

3. Simulated Experiment Results

This section presents a simulation example (i.e., mixture normal model). The main purpose of this example is to demonstrate that the MTMSA algorithm could compute the optimization problem in the case of multiple local maxima and outperform its counterparts in terms of accuracy and efficiency. All results are obtained using R language (version X64 2.3.4) and MATLAB (version R2017a) on a Dell OptiPlex7020MT desktop computer with Intel(R) Core(TM) i7-4790 CPU @ 3.6 GHz, RAM 8.00 GB, and Windows 7 Ultimate with Service Pack 1 (x64). The R and MATLAB codes in this work are available upon request to the corresponding author.

In this simulation example, we consider a two-dimensional multimodal mixture normal model modified from [27, 28]. In this model, the objective function is the probability density function, which is the combination of 20 normal models

$$f(\mathbf{x}) \propto \sum_{i=1}^{20} \frac{\omega_i}{2\pi\sigma_i^2} \exp^{-(1/2\sigma_i^2)(\mathbf{x}-\boldsymbol{\mu}_i)'(\mathbf{x}-\boldsymbol{\mu}_i)}, \tag{5}$$

where $\sigma_1 = \sigma_2 = \cdots = \sigma_{20} = 0.1$ and $(\omega_1, \omega_2, \ldots, \omega_{20})$, which are the weights of the 20 normal models, are chosen to be the arithmetic progression from 1 to 5, except the last one $\omega_{20} = 10$. The 20 mean vectors are independently sampled from the uniform distribution from $[0, 10]$.

Figure 1 illustrates the mesh and contour plots of (5), which contains 20 modes in this objective function. This example poses a serious challenge for optimization because many classical optimization methods may converge on the local optimum in this multimodal example. Clearly, the global maximum point is the last mode $(0.183, 7.501)$ with the maximum value of 2.449.

Four methods are used to find the global maximum point of this optimization problem: the NM method, modified QN method, SA algorithm, and MTMSA algorithm. The NM and QN methods are commonly applied numerical optimization algorithms, and the QN method allows box constraints. The SA and its improved version, i.e., the MTMSA algorithm, are stochastic optimization algorithms. Apart from the minimum temperature T_{min}, another commonly used parameter that controls the degree of decreasing temperature is N_{temper}:

$$T_k = T_{max} \cdot \alpha^{k-1}, \quad k = 1, 2, \ldots, N_{temper}. \tag{6}$$

For the SA algorithm, we set the degree of decreasing temperature $N_{temper} = 75$, the starting temperature $T_{max} = 10$, the decreasing parameter $\alpha = 0.9$, the length of the Markov chain $N_{mc} = 100$, and the proposal variance of the Metropolis algorithm $v_{pro} = 2$. For the MTMSA algorithm, we set the number of multiple-tries $N_{mtm} = 100$. The other parameters are $N_{temper} = 25$, $\alpha = 0.8$, $T_{max} = 1$, $N_{mc} = 100$, and $v_{pro} = 2$. With different N_{temper} (75 and 25) and α (0.9 and 0.8, respectively) values, the SA and MTMSA algorithms have similar T_{min}.

We tested these four algorithms (NM, QN, SA, and MTMSA algorithms) to compute the optimization problem and repeated this computation 50 times. The computation results are summarized in Figure 2 and Table 1.

The mean value, standard deviation (sd), mean square error (MSE), total CPU time (in seconds), and average CPU time for one accurate result (in seconds) of the results from different algorithms are summarized in Table 1, where $mean = (1/R)\sum_{i=1}^{R} V_i$, $sd = \sqrt{(1/(R-1))\sum_{i=1}^{R}(V_i - mean)^2}$, $MSE = (1/R)\sum_{i=1}^{R}(V_i - V_e)^2$, and $V_e = 2.449$ is the exact maximum value in the model (5). Figure 2 illustrates the boxplot of 50 computation results from these four algorithms.

Suffering from the "local-trap" problem, NM and QN algorithms cannot find the global maximum successfully in 50 computations (they often find other local modes in (5)). Compared with the MTMSA algorithm, the SA algorithm uses a slowly decreasing temperature schedule ($N_{temper} = 75$) and consumes more CPU time. However, only 10 results of 50 repetitions from the SA algorithm converge to the

Table 1: Computation results (mean, sd, MSE, consumed total CPU time, and average CPU time (in seconds)) of the 50 computations from different algorithms.

	DM	QN	SA	MTMSA
mean	0.3447	0.0311	1.7523	2.4392
sd	0.4563	0.1355	0.6427	0.0086
MSE	4.6319	5.8635	0.8901	0.0001
total CPU time	4.1732	0.9121	1305.6	528.1
average CPU time	$+\infty$	$+\infty$	130.56	10.562

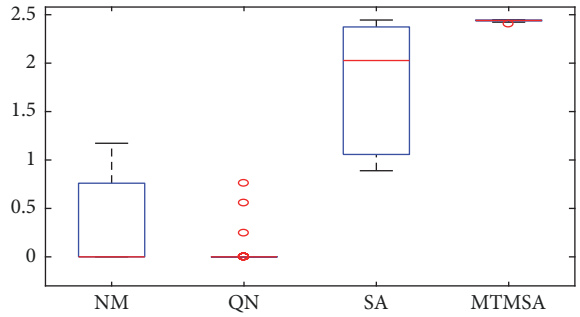


Figure 2: Boxplot of 50 results computed from the NM method, QN method, SA algorithm, and MTMSA algorithm.

global maximum point $(0.183, 7.501)$, and the mean of the SA algorithm is 1.7523. By contrast, the MTMSA algorithm functions with a rapidly decreasing temperature schedule. The MTMSA algorithm consumes minimal CPU time (only about 8 min), but it yields highly accurate results (all 50 results converge to the global maximum). Furthermore, the MTMSA algorithm only needs approximately 10 seconds to compute one accurate result, whereas the SA algorithm requires about 130 seconds. All results from NM and QN algorithms suffer from the "local-trap" problem.

We compared the differences in the decreasing temperature schedules used in SA and MTMSA algorithms. "The slower the better" was found to be applicable to the decreasing temperature schedules. A rapidly decreasing temperature schedule may result in the "local-trap" problem. In the next simulation, we set the temperature schedules to decrease from 10 to 5×10^{-3}, and the length of decreasing temperature was set to 500, 75 and 25 for SA and MTMSA algorithms. Each computation was repeated 50 times.

The length of decreasing temperature of the SA algorithm is set to 75, 500, and denoted as the SA1 and SA2 algorithm, respectively. The SA2 algorithm shows the slowest decreasing schedule. It uses 500 steps to drop from the highest temperature to the lowest one. Thus almost all 50 results converge to the global maximum (about 94% percent of computation results escape from local optima and reaches the global maximum). The SA1 algorithm uses a rapidly decreasing schedule, and only about half of the 50 results converge to the global maximum (about 54% percent of computation results escape from local optima and reaches the global maximum). By contrast, the MTMSA algorithm only uses 25 steps in decreasing temperature, but all of the 50 results converge to the global maximum.

Table 2: Biochemical oxygen demand versus time.

Time (days)	BOD (mg/I)
1	8.3
2	10.3
3	19.0
4	16.0
5	15.6
7	19.8

Figure 3 shows the decreasing schedules and convergence paths of the three algorithms. We find that when the temperature decreases to about 0.02 (corresponding to the 50th, 400th, and 20th steps in SA1, SA2, and MTMSA), all sample paths from the three algorithms converge to their local and global optima. All the sample paths of MTMSA converge to the global optima, and lots of sample paths of SA1 and SA2 converge to the local optima because the average sample path of MTMSA in Figure 3 is the highest and at the level about 2.43. The MTMSA algorithm uses the rapidly decreasing schedule and achieves the fastest convergence rate. Therefore, the MTMSA algorithm is the most efficient and accurate in this simulation example.

4. Real Data Examples

4.1. Bayesian Analysis Using MTMSA. In this section, we illustrate the application of the MTMSA algorithm in Bayesian analysis with real data from [29]. In this example, we fit a nonlinear model derived from exponential decay

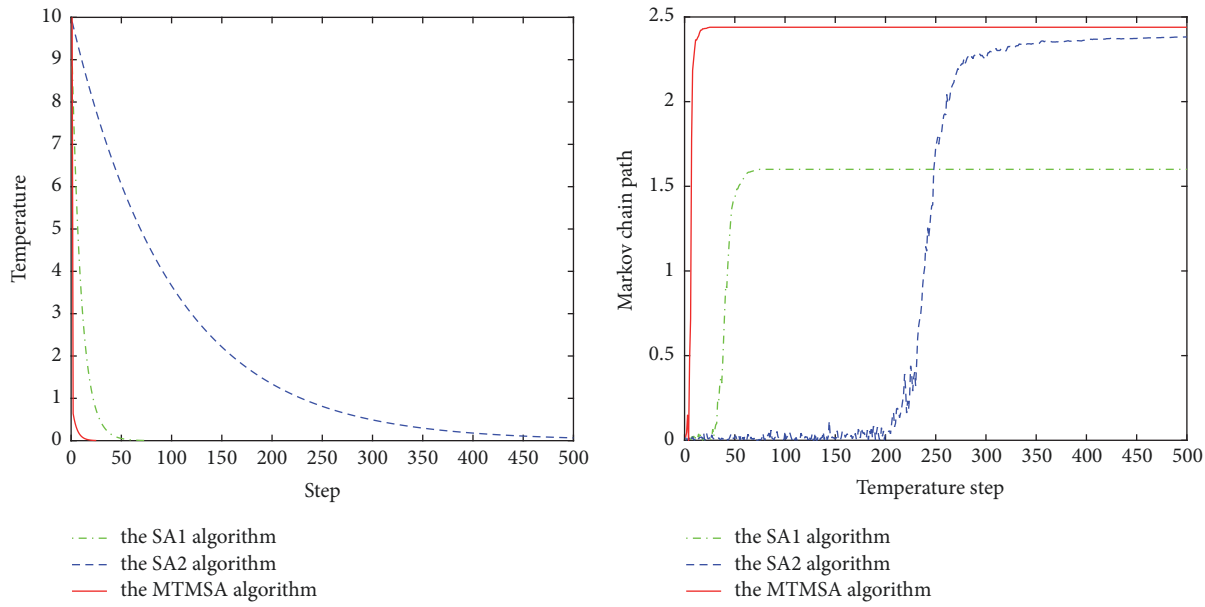
$$y_i = \theta_1 \left(1 - \exp\left\{-\theta_2 x_i\right\}\right) + \varepsilon_i, \quad \varepsilon_i \sim N\left(0, \sigma^2\right), \tag{7}$$

with a fixed rate that is constant to a real data set [30] (Table 2).

The variables BOD (mg/I)) and time (days) in Table 2 are the response and control variables in model (7) (denoted as the BOD problem) with a constant variance σ^2 for independent normal errors. The likelihood for the BOD problem is

$$L\left(\theta_1, \theta_2, \sigma^2 \mid X, Y\right) \propto \exp\left\{-6\log\sigma - \frac{1}{2}\frac{\sum_{i=1}^{6}\left(y_i - \theta_1\left(1 - \exp\left\{-\theta_2 x_i\right\}\right)\right)^2}{\sigma^2}\right\}, \tag{8}$$

where $X = (x_1, \ldots, x_6)$ and $Y = (y_1, \ldots, y_6)$.



(a) The decreasing temperature schedules

(b) The convergence paths of different algorithms

Figure 3: Decreasing temperature schedules (a) and convergence paths (b) of the SA1 algorithm, SA2 algorithm, and MTMSA algorithm. The convergence paths are the average of 50 paths.

Table 3: Computation results (mean (10^{-5}), sd (10^{-5})) in special temperature steps (1, 5, 10, 15, 20, 25) from 20 repetitions.

	step 1	step 5	step 10	step 15	step 20	step 25
mean	1.05	1.39	0.91	3.92	145	148
sd	3.57	5.16	2.20	12.2	3.04	0.24

While choosing the flat prior for the parameters σ^2 and (θ_1, θ_2) (i.e., the uniform distribution in $(0, +\infty)$ and $[-20, 50] \times [-2, 6]$, respectively) and integrating out σ^2, we obtain the following (improper) posterior distribution of (θ_1, θ_2):

$$p(\theta_1, \theta_2 \mid X, Y) \propto \left[\sum_{i=1}^{6} \left(y_i - \theta_1 \left(1 - \exp\{-\theta_2 x_i\}\right)\right)^2\right]^{-2} \cdot I_{[-20,50]\times[-2,6]}(\theta_1, \theta_2), \quad (9)$$

where

$$I_{[-20,50]\times[-2,6]}(\theta_1, \theta_2) = \begin{cases} 1, & (\theta_1, \theta_2) \in [-20, 50] \times [-2, 6] \\ 0, & (\theta_1, \theta_2) \notin [-20, 50] \times [-2, 6]. \end{cases} \quad (10)$$

For a Bayesian analysis, one often treats the parameters (θ_1, θ_2) as random variables. In this work, we use the posterior distribution of (θ_1, θ_2) for their statistical inference and use the posterior mode of (9) as the estimation of (θ_1, θ_2), which coincides with the maximum likelihood estimation. The Bayesian statistical inference of the parameters (θ_1, θ_2) is translated to the global optimization problem in $[-20, 50] \times [-2, 6]$.

$$\sup_{(\theta_1,\theta_2)\in[-20,50]\times[-2,6]} p(\theta_1, \theta_2 \mid X, Y). \quad (11)$$

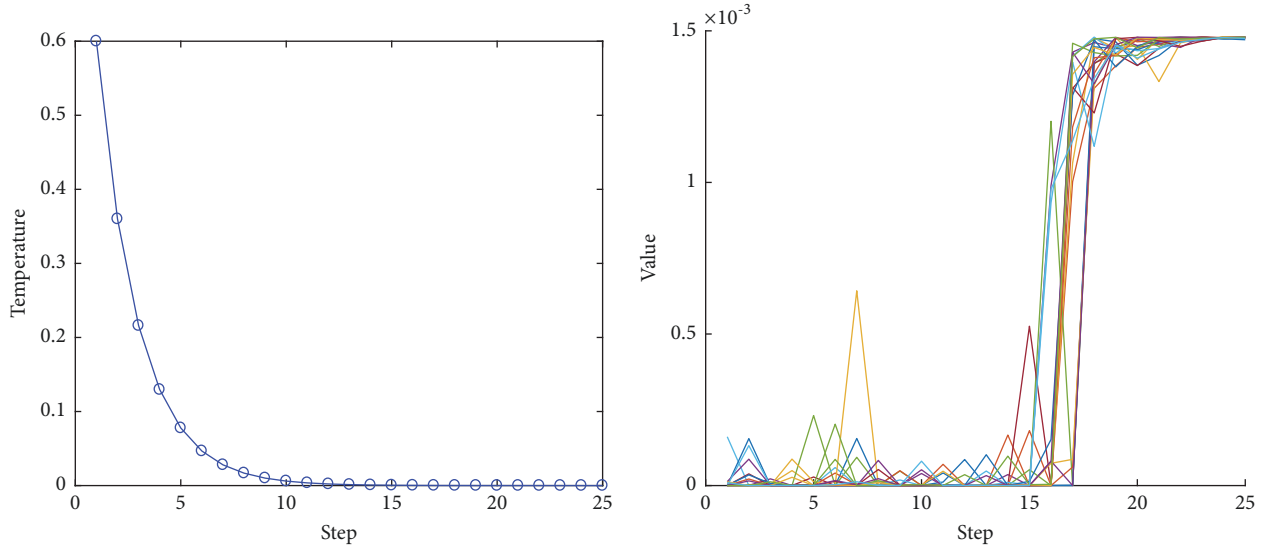
In addition, we use the MTMSA algorithm to compute the global optimization problem (11). The parameters of the MTMSA algorithm are set to be $N_{mtm} = 20$, $N_{temper} = 25, \alpha = 0.6, T_{max} = 1$, and $N_{mc} = 1000$. The computation is then repeated 20 times. Figure 4 and Table 3 illustrate the decreasing temperature schedule and the convergence paths of 20 repetitions from the MTMSA algorithm. After 20 steps, all 20 computation paths become convergent to 1.48×10^{-3}, which has the largest mean and smallest sd.

Figure 5 shows the mesh (a) and contour (b) plots of the posterior distribution (9). Figure 6 and Table 4 show the locations of the scatters (θ_1, θ_2) from 20 repetitions at different temperature steps. With the temperature decreasing from 0.6 to 2.8×10^{-6}, all scatters converge to the optimization point (19.15, 0.53).

4.2. Half-Space Depth Computation Using MTMSA. As a powerful tool for nonparametric multivariate analysis, half-space depth (HD also known as Tukey depth) has been eliciting increased interest since it was introduced by Tukey [31, 32]. HD, which extends univariate order-related statistics to multivariate settings, provides a center-outward ordering

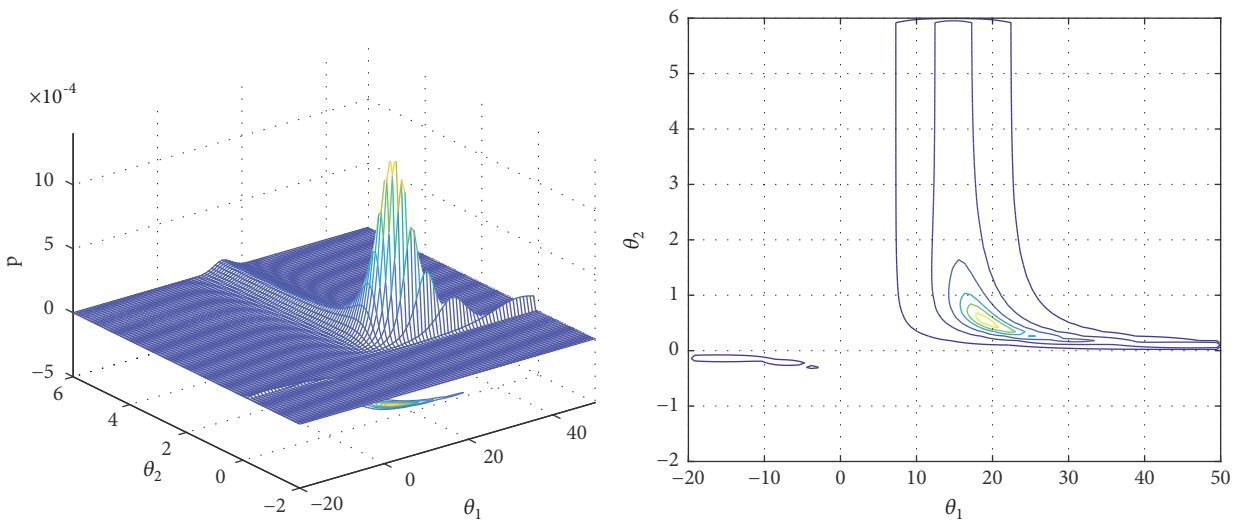
Table 4: Location results of (θ_1, θ_2) at different temperature levels.

Level	0.6	2.8×10^{-4}	1.6×10^{-4}	2.8×10^{-6}
mean	(20.22, 1.80)	(16.63, 1.06)	(18.25, 1.05)	(19.15, 0.53)
sd	(21.71, 2.39)	(17.64, 1.92)	(9.58, 1.56)	(0.11, 0.01)



(a) (b)

Figure 4: The decreasing temperature schedule (a) and the convergence paths of 20 repetitions (b) from the MTMSA algorithm.



(a) Mesh plot of posterior distribution (b) Contour plot of posterior distribution

Figure 5: Exact mesh (a) and contour (b) plots of the posterior distribution (9).

of multivariate samples and visualizes data in high dimensional cases [33, 34]. However, the computation of HD is challenging, and the exact algorithm is often inefficient, especially when the dimension is high [35]. In this subsection, we use MTMSA to compute HD and compared MTMSA with other approximated and exact algorithms.

Given a sample data set of size n $\mathbf{X}^n = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n\}$ in $\mathbb{R}^d$, $\mathbf{x}$ is a point in $\mathbb{R}^d$, and the HD of $\mathbf{x}$ with respect to (w.r.t.) $\mathbf{X}^n$ is defined by

$$HD\left(\mathbf{x}, \mathbf{X}^n\right) = \min_{\mathbf{u} \in \mathbb{S}^{d-1}} \frac{1}{n} \# \left\{ i \mid \mathbf{u}^T \mathbf{X}_i \geq \mathbf{u}^T \mathbf{x},\ i \in \mathcal{N} \right\}, \quad (12)$$

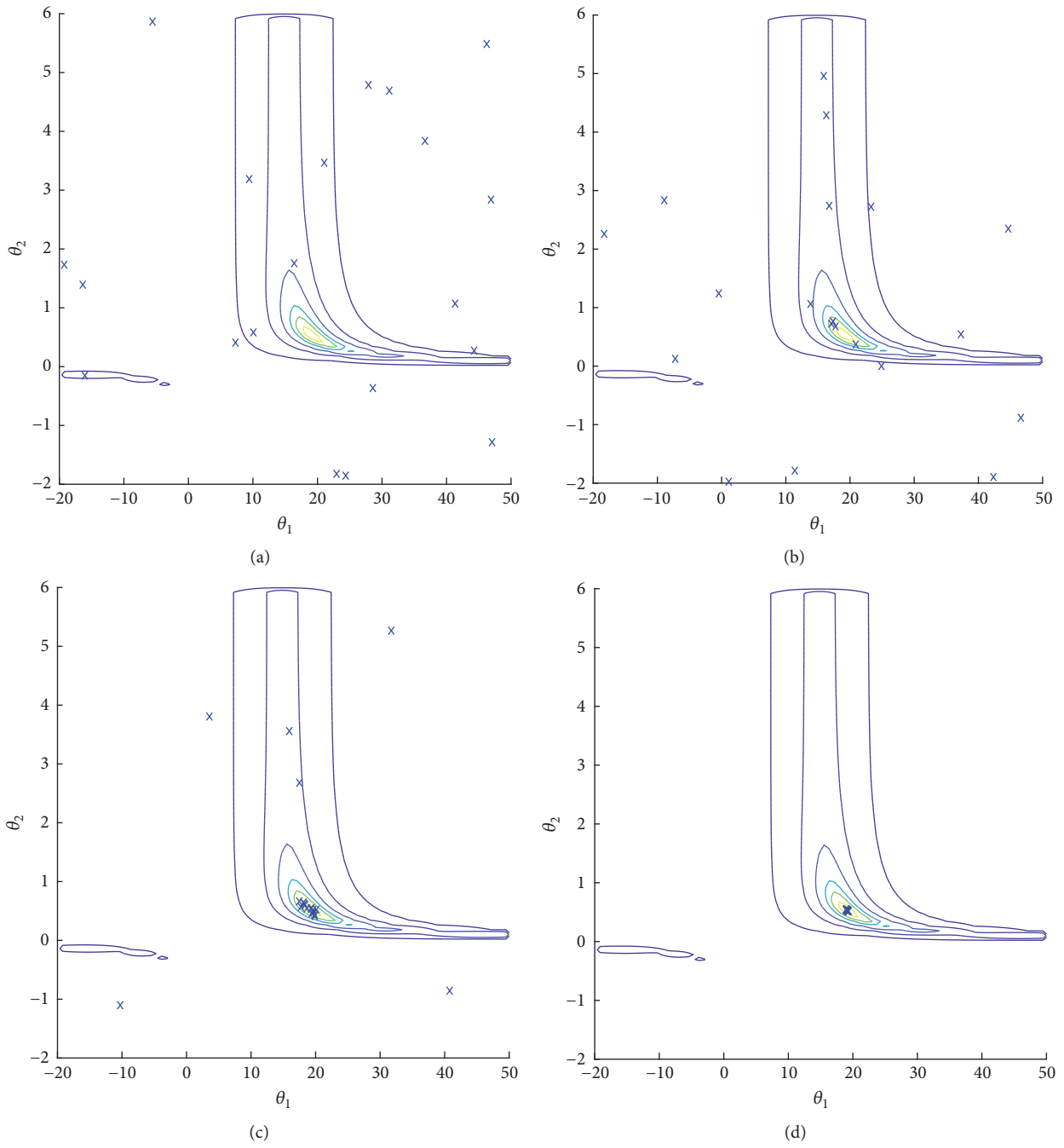


Figure 6: Locations of (θ_1, θ_2) from 20 repetitions at different temperature steps $(0.6, 2.8 \times 10^{-4}, 1.6 \times 10^{-4}, 2.8 \times 10^{-6},)$.

where $\mathbb{S}^{d-1} = \{u \in \mathbb{R}^d \mid \|u\| = 1\}$, $\mathcal{N} = \{1, 2, \ldots, n\}$, and $\#\{\cdot\}$ denotes the counting measure. Then, the computation of HD (12) is a global optimization problem in $\mathbb{S}^{d-1}$.

Next, we considered a concrete data set (Table 6) obtained from [35] and can be found in the Records Office of the Laboratory School of the University of Chicago. The original data consisted of 64 subjects' scores obtained from eighth-grade levels to eleventh-grade levels. Then, we compared MTMSA with three approximated algorithms (NM, QN, and SA) and the exact algorithm from [35] for the HD computation of the first data point w.r.t. the data set.

We tested two sets of parameters for the SA algorithm. The first is $N_{temper} = 20$, $N_{mc} = 50$, $T_{max} = 1$, and $a = 0.7$ and denoted as the SA1 algorithm. The second one is $N_{temper} = 20$, $N_{mc} = 200$, $T_{max} = 1$, and $a = 0.7$ and denoted as the SA2 algorithm. For the MTMSA algorithm, we set the parameter to be $N_{temper} = 20$, $m = 100$, $N_{mc} = 30$, $T_{max} = 1$, and $a = 0.7$. The three algorithms (SA1, SA2, and MTMSA) use the same decreasing temperature schedule. Then, we used the six algorithms (exact, NM, QN, SA1, SA2, and MTMSA) for this computation and repeated the computation 50 times. Figure 7 and Table 5 show the computation results.

Table 5: Computation results (mean, sd, MSE, consumed total CPU time, and average CPU time (in seconds)) of the 50 computations from different algorithms.

	exact	NM	QN	SA1	SA2	MTMSA
mean	0.2344	0.3653	0.3841	0.2609	0.2425	0.2344
sd	0	0.0519	0.0485	0.0243	0.0079	0
MSE	0	0.0199	0.0247	0.0013	0.0001	0
total CPU time	2450	0.06	0.05	5.7410	23.103	15.87
average CPU time	49	$+\infty$	$+\infty$	0.9570	0.9626	0.3174

Table 6: Concrete data set.

subject	Grade 8	Grade 9	Grade 10	Grade 11
1	1.75	2.60	3.76	3.68
2	0.90	2.47	2.44	3.43
3	0.80	0.93	0.40	2.27
4	2.42	4.15	4.56	4.21
5	−1.31	−1.31	−0.66	−2.22
6	−1.56	1.67	0.18	2.33
7	1.09	1.50	0.52	2.33
8	−1.92	1.03	0.50	3.04
9	−1.61	0.29	0.73	3.24
10	2.47	3.64	2.87	5.38
11	−0.95	0.41	0.21	1.82
12	1.66	2.74	2.40	2.17
13	2.07	4.92	4.46	4.71
14	3.30	6.10	7.19	7.46
15	2.75	2.53	4.28	5.93
16	2.25	3.38	5.79	4.40
17	2.08	1.74	4.12	3.62
18	0.14	0.01	1.48	2.78
19	0.13	3.19	0.60	3.14
20	2.19	2.65	3.27	2.73
21	−0.64	−1.31	−0.37	4.09
22	2.02	3.45	5.32	6.01
23	2.05	1.80	3.91	2.49
24	1.48	0.47	3.63	3.88
25	1.97	2.54	3.26	5.62
26	1.35	4.63	3.54	5.24
27	−0.56	−0.36	1.14	1.34
28	0.26	0.08	1.17	2.15
29	1.22	1.41	4.66	2.62
30	−1.43	0.80	−0.03	1.04
31	−1.17	1.66	2.11	1.42
32	1.68	1.71	4.07	3.30
33	−0.47	0.93	1.30	0.76
34	2.18	6.42	4.64	4.82
35	4.21	7.08	6.00	5.65
36	8.26	9.55	10.24	10.58
37	1.24	4.90	2.42	2.54
38	5.94	6.56	9.36	7.72
39	0.87	3.36	2.58	1.73
40	−0.09	2.29	3.08	3.35
41	3.24	4.78	3.52	4.84
42	1.03	2.10	3.88	2.81
43	3.58	4.67	3.83	5.19
44	1.41	1.75	3.70	3.77
45	−0.65	−0.11	2.40	3.53
46	1.52	3.04	2.74	2.63
47	0.57	2.71	1.90	2.41
48	2.18	2.96	4.78	3.34
49	1.10	2.65	1.72	2.96
50	0.15	2.69	2.69	3.50
51	−1.27	1.26	0.71	2.68
52	2.81	5.19	6.33	5.93
53	2.62	3.54	4.86	5.80
54	0.11	2.25	1.56	3.92
55	0.61	1.14	1.35	0.53
56	−2.19	−0.42	1.54	1.16
57	1.55	2.42	1.11	2.18
58	0.04	0.50	2.60	2.61
59	3.10	2.00	3.92	3.91
60	−0.29	2.62	1.60	1.86
61	2.28	3.39	4.91	3.89
62	2.57	5.78	5.12	4.98
63	−2.19	0.71	1.56	2.31
64	−0.04	2.44	1.79	2.64

Figure 7 and Table 5 show that the exact algorithm consumed the most CPU time (about 2450 seconds) and obtained exact computation results (0.2344). MTMSA also obtained the exact results but consumed only 15.87 seconds. The SA algorithms (SA1 and SA2) consumed suitable CPU time (5.741 and 23.103 seconds, respectively) but obtained only 6 and 24 exact results, respectively. The results of NM and QN fell into the local optima because all of them were larger than the exact result. With regard to the average CPU time, MTMSA used only 0.3174 for the computation of one exact result, which is the least amount of time compared with the time for the other exact and approximated algorithms. Hence, MTMSA outperformed the other algorithms in this experiment example.

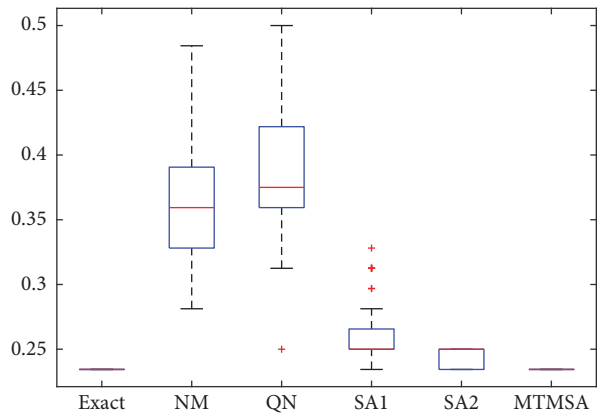


Figure 7: Boxplot of the results computed from the exact, NM, QN, SA1, SA2, and MTMSA algorithms.

5. Conclusions

We developed the MTMSA algorithm for global optimization problems in the fields of mathematical/biological sciences, engineering, Bayesian data analysis, operational research, life sciences, and so on. The MTMSA algorithm is a combination of the SA algorithm and the MTM algorithm. Using simulated and real data examples, it demonstrated that, relative to the QN and SA algorithm, the MTMSA algorithm can function with a rapidly decreasing cooling schedule while guaranteeing that the global energy minima are reached.

Several directions can be taken for future work. First, combined with the quasi-Monte Carlo method, the low-discrepancy sequences and experimental design [36–39] can be used to accelerate the convergence of the SA algorithm. Second, aside from the MTM algorithm, the MTMSA algorithm can also be implemented with several parallel interacting Markov chains to improve the SA algorithm by making full use of modern multicore computer [40, 41]. Third, we anticipate that a parallel SA algorithm can be used efficiently for variable selection in high dimensional cases [42–45] because the variable selection problem is a special case of the optimization problem. Finally, data depth [32, 33, 35, 46] is an important tool for multidimensional data analysis, but the computation of data depth in high dimensional cases is challenging. The example of half-space depth computation in Section 4 shows the advantage of the MTMSA algorithm in low dimensional case. Hence, we believe that the MTMSA algorithm can be successfully applied to compute highly complex data depths (e.g., projection and regression depths) in high dimensional cases. Further analysis along these directions would be interesting.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

The research was partially supported by the National Natural Science Foundation of China (11501320, 71471101, and 11426143), the Natural Science Foundation of Shandong Province (ZR2014AP008), and the Natural Science Foundation of Qufu Normal University (bsqd20130114).

References

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

[2] F. Liang, C. Liu, and R. J. Carroll, *Advanced Markov Chain Monte Carlo Methods: Learning From Past Samples*, John Wiley & Sons, 2011.

[3] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *American Association for the Advancement of Science: Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[4] Y. Wang, L. Qi, S. Luo, and Y. Xu, "An alternative steepest direction method for the optimization in evaluating geometric discord," *Pacific Journal of Optimization*, vol. 10, no. 1, pp. 137–149, 2014.

[5] C. Wang and Y. Wang, "A superlinearly convergent projection method for constrained systems of nonlinear equations," *Journal of Global Optimization*, vol. 44, no. 2, pp. 283–296, 2009.

[6] Y. Wang, L. Caccetta, and G. Zhou, "Convergence analysis of a block improvement method for polynomial optimization over unit spheres," *Numerical Linear Algebra with Applications*, vol. 22, no. 6, pp. 1059–1076, 2015.

[7] L. Qi, X. Tong, and Y. Wang, "Computing power system parameters to maximize the small signal stability margin based on min-max models," *Optimization and Engineering*, vol. 10, no. 4, pp. 465–476, 2009.

[8] H. Chen, Y. Chen, G. Li, and L. Qi, "A semidefinite program approach for computing the maximum eigenvalue of a class of structured tensors and its applications in hypergraphs and copositivity test," *Numerical Linear Algebra with Applications*, vol. 25, no. 1, 2018.

[9] G. Wang and X. X. Huang, "Levitin-Polyak well-posedness for optimization problems with generalized equilibrium constraints," *Journal of Optimization Theory and Applications*, vol. 153, no. 1, pp. 27–41, 2012.

[10] G. Wang, "Levitin-Polyak well-posedness for vector optimization problems with generalized equilibrium constraints," *Pacific Journal of Optimization*, vol. 8, no. 3, pp. 565–576, 2012.

[11] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, 1984.

[12] R. A. Holley, S. Kusuoka, and D. W. Stroock, "Asymptotics of the spectral gap with applications to the theory of simulated annealing," *Journal of Functional Analysis*, vol. 83, no. 2, pp. 333–347, 1989.

[13] F. Liang, C. Liu, and R. J. Carroll, "Stochastic approximation in Monte Carlo computation," *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 305–320, 2007.

[14] F. Liang, Y. Cheng, and G. Lin, "Simulated stochastic approximation annealing for global optimization with a square-root cooling schedule," *Journal of the American Statistical Association*, vol. 109, no. 506, pp. 847–863, 2014.

[15] G. Karagiannis, B. A. Konomi, G. Lin, and F. Liang, "Parallel and interacting stochastic approximation annealing algorithms for global optimisation," *Statistics and Computing*, vol. 27, no. 4, pp. 927–945, 2017.

[16] J. S. Liu, F. Liang, and W. H. Wong, "The multiple-try method and local optimization in metropolis sampling," *Journal of the American Statistical Association*, vol. 95, no. 449, pp. 121–134, 2000.

[17] R. Casarin, R. Craiu, and F. Leisen, "Interacting multiple try algorithms with different proposal distributions," *Statistics and Computing*, vol. 23, no. 2, pp. 185–200, 2013.

[18] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[19] W. Shao, G. Guo, F. Meng, and S. Jia, "An efficient proposal distribution for metropolis-hastings using a b-splines technique," *Computational Statistics & Data Analysis*, vol. 57, pp. 465–478, 2013.

[20] W. Shao and Y. Zuo, "Simulated annealing for higher dimensional projection depth," *Computational Statistics & Data Analysis*, vol. 56, no. 12, pp. 4026–4036, 2012.

[21] W. Shao, G. Guo, G. Zhao, and F. Meng, "Simulated annealing for the bounds of kendall's tau and spearman's rho," *Journal of Statistical Computation and Simulation*, vol. 84, no. 12, pp. 2688–2699, 2014.

[22] Y. Luo, B. Zhu, and Y. Tang, "Simulated annealing algorithm for optimal capital growth," *Physica A: Statistical Mechanics and its Applications*, vol. 408, pp. 10–18, 2014.

[23] O. S. Sarıyer and C. Güven, "Simulated annealing algorithm for optimal capital growth," *Physica A: Statistical Mechanics and its Applications*, vol. 408, pp. 10–18, 2014.

[24] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[25] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[26] R. H. Byrd, P. Lu, J. Nocedal, and C. Y. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[27] S. C. Kou, Q. Zhou, and W. H. Wong, "Equi-energy sampler with applications in statistical inference and statistical mechanics," *The Annals of Statistics*, vol. 34, no. 4, pp. 1581–1652, 2006.

[28] F. Liang and W. H. Wong, "Real-parameter evolutionary Monte Carlo with applications to Bayesian mixture models," *Journal of the American Statistical Association*, vol. 96, no. 454, pp. 653–666, 2001.

[29] C. Ritter and M. A. Tanner, "Facilitating the Gibbs sampler: The Gibbs stopper and the Griddy–Gibbs sampler," *Journal of the American Statistical Association*, vol. 87, no. 419, pp. 861–868, 1992.

[30] D. M. Bates and D. G. Watts, *Nonlinear Regression Analysis and Its Applications*, John Wiley & Sons, New York, NY, USA, 1988.

[31] J. W. Tukey, "Mathematics and the picturing of data," *Proceedings of the International Congress of Mathematicians*, vol. 2, pp. 523–531, 1975.

[32] Y. Zuo and R. Serfling, "General notions of statistical depth function," *The Annals of Statistics*, vol. 28, no. 2, pp. 461–482, 2000.

[33] X. Liu, Y. Zuo, and Q. Wang, "Finite sample breakdown point of Tukey's halfspace median," *Science China Mathematics*, vol. 60, no. 5, pp. 861–874, 2017.

[34] X. Liu, "Fast implementation of the Tukey depth," *Computational Statistics*, vol. 32, no. 4, pp. 1395–1410, 2017.

[35] X. Liu and Y. Zuo, "Computing halfspace depth and regression depth," *Communications in Statistics—Simulation and Computation*, vol. 43, no. 5, pp. 969–985, 2014.

[36] Z. Li, S. Zhao, and R. Zhang, "On general minimum lower order confounding criterion for s-level regular designs," *Statistics & Probability Letters*, vol. 99, pp. 202–209, 2015.

[37] J. Wang, Y. Yuan, and S. Zhao, "Fractional factorial split-plot designs with two- and four-level factors containing clear effects," *Communications in Statistics—Theory and Methods*, vol. 44, no. 4, pp. 671–682, 2015.

[38] S. Zhao, D. K. Lin, and P. Li, "A note on the construction of blocked two-level designs with general minimum lower order confounding," *Journal of Statistical Planning and Inference*, vol. 172, pp. 16–22, 2016.

[39] S.-L. Zhao and Q. Sun, "On constructing general minimum lower order confounding two-level block designs," *Communications in Statistics—Theory and Methods*, vol. 46, no. 3, pp. 1261–1274, 2017.

[40] G. Guo, W. You, G. Qian, and W. Shao, "Parallel maximum likelihood estimator for multiple linear regression models," *Journal of Computational and Applied Mathematics*, vol. 273, pp. 251–263, 2015.

[41] G. Guo, W. Shao, L. Lin, and X. Zhu, "Parallel tempering for dynamic generalized linear models," *Communications in Statistics—Theory and Methods*, vol. 45, no. 21, pp. 6299–6310, 2016.

[42] M. Wang and G.-L. Tian, "Robust group non-convex estimations for high-dimensional partially linear models," *Journal of Nonparametric Statistics*, vol. 28, no. 1, pp. 49–67, 2016.

[43] M. Wang, L. Song, and G.-l. Tian, "Scad-penalized least absolute deviation regression in high-dimensional models," *Communications in Statistics—Theory and Methods*, vol. 44, no. 12, pp. 2452–2472, 2015.

[44] G.-L. Tian, M. Wang, and L. Song, "Variable selection in the high-dimensional continuous generalized linear model with current status data," *Journal of Applied Statistics*, vol. 41, no. 3, pp. 467–483, 2014.

[45] M. Wang and X. Wang, "Adaptive Lasso estimators for ultrahigh dimensional generalized linear models," *Statistics & Probability Letters*, vol. 89, pp. 41–50, 2014.

[46] P. J. Rousseeuw and M. Hubert, "Regression depth," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 388–433, 1999.