MOMCMC: AN EFFICIENT MONTE CARLO METHOD FOR MULTI-OBJECTIVE SAMPLING OVER REAL PARAMETER SPACE

Yaohang Li
Department of Computer Science
Old Dominion University
Norfolk, VA 23529, USA
yaohang@cs.odu.edu

Abstract. In this paper, we present a new population-based Monte Carlo method, so-called MOMCMC (Multi-Objective Markov Chain Monte Carlo), for sampling in the presence of multiple objective functions in real parameter space. The MOMCMC method is designed to address the "multi-objective sampling" problem, which is not only of interest in exploring diversified solutions at the Pareto optimal front in the function space of multiple objective functions, but also those near the front. MOMCMC integrates Differential Evolution (DE) style crossover into Markov Chain Monte Carlo (MCMC) to adaptively propose new solutions from the current population. The significance of dominance is taken into consideration in MOMCMC's fitness assignment scheme while balancing the solution's optimality and diversity. Moreover, the acceptance rate in MOMCMC is used to control the sampling bandwidth of the solutions near the Pareto optimal front. As a result, the computational results of MOMCMC with the high-dimensional ZDT benchmark functions demonstrate its efficiency in obtaining solution samples at or near the Pareto optimal front. Compared to MOSCEM (Multiobjective Shuffled Complex Evolution Metropolis), an existing Monte Carlo sampling method for multi-objective optimization, MOMCMC exhibits significantly faster convergence to the Pareto optimal front. Furthermore, with small population size, MOMCMC also shows effectiveness in sampling complicated multi-objective function space.

Key words. Markov Chain Monte Carlo, Multi-Objective Sampling, Pareto Optimal Front, Convergence

1. Introduction

In many scientific and engineering applications, two or more potentially conflicting objectives coexist, where optimal solutions in the presence of tradeoffs among these objectives are of particular interest. The collection of these optimal solutions is called the Pareto optimal front [16]. Correspondingly, the process of systematically optimizing multiple objective functions to obtain a good coverage of the Pareto-optimal solutions is referred to as Multi-Objective Optimization (MOO). A variety of evolutionary algorithms, such as NSGA-II (Nondominated Sorted Genetic Algorithm II) [1], SPEA-II (Strength Pareto Evolutionary Algorithm II) [2], PAES (Pareto-Archived Evolution Strategy) [3], and many others have been developed to address MOO problems and have been effectively used in a wide variety of applications. Deb [1] and Coello Coello et al. [28] provide excellent surveys and introductions to the early evolutionary algorithms for solving MOO problems. More recently, multi-objective evolutionary algorithms with enhanced strategies, such as ϵ-domination based multi-objective evolutionary algorithm [29], multi-objective evolutionary algorithm based on decomposition (MOEA/D) [30], hypervolume-based multi-objective optimization [31], auto-adaptive multi-objective evolutionary computing [32], and AMALGAM (multialgorithm, genetically adaptive multiobjective) [33], have demonstrated further efficiency improvement in various MOO problems.

Nevertheless, in many applications, one may not only be interested in optimal solutions, but also in their uncertainty distributions. For example, in drug design, chemical compounds near the optimum of the design objectives are often considered to be good candidates for further screening due to the fact that the design objectives are usually vague and uncertain [4]. Protein structure modeling is another application example. One desires to sample the structurally diversified models close to the optima of the simplified and potentially inaccurate backbone scoring (energy) functions with reduced protein representation to discover near-native conformations [5, 26]. Therefore, different from MOO, this paper is particularly interested in the "multi-objective sampling problem," i.e., to explore the diversified solutions at the Pareto optimal front as well as those near the front in the function space of multiple objective functions.

Most available Markov Chain Monte Carlo (MCMC) methods, such as Metropolis-Hastings [6, 7], Simulated Annealing [8], Simulated Tempering [9], configuration-biased Monte Carlo [10], and Parallel Tempering [11], are applied to applications with only a single objective (energy) function. In 2002, Vrugt et al. extended MCMC methods to applications with multiple objective functions by taking advantage of the concept of Pareto dominance, where the resulting Multiobjective Shuffled Complex Evolution Metropolis (MOSCEM) method [12] has demonstrated certain effectiveness and efficiency in multi-objective optimization of hydrologic models. However, recent study [13] has shown that, in a set of standardized test functions and in a few hydrologic calibration cases, the MOSCEM method is inferior to genetic algorithm-based multi-objective optimization algorithms, such as NSGA-II [1] and SPEA-II [2], in finding solutions at the Pareto optimal front. This is based on measurement of both the capability of reaching Pareto optimal solutions and the computational time. One main reason is that the stochastic search operator in MOSCEM depends on the prior distribution, which requires the covariance calculation of the parameters in the population. Due to this, the stochastic search in MOSCEM using complex shuffling "does not scale well for problems of increasing size and/or difficulty." [13]

In this paper, we present a Multi-Objective Markov Chain Monte Carlo (MOMCMC) method for sampling the optimal as well as near-optimal solutions in the presence of multiple objective functions over real parameter space. We want to emphasize that MOMCMC intends to tackle the multi-objective sampling problem, i.e., we are not only interested in the Pareto optimal solutions, but also the near Pareto optimal solutions. This is different from the MOO problem where only Pareto optimal solutions are of interest. MOMCMC is a population-based MCMC approach which incorporates several novel improvements overcoming the deficiencies in MOSCEM. First of all, a Differential Evolution (DE) [14] style crossover among solutions in a population is used to implement the proposal function in MCMC with appropriate scale and orientation. Secondly, a fitness assignment approach is developed to distinguish dominated and non-dominated solutions and to measure the significance of dominance while preserving solution diversity. Thirdly, the acceptance rate in MOMCMC is used as a tuner to control the sampling bandwidth of the solutions near the Pareto optimal front. We use a set of real parameter benchmark functions for multi-objective optimization to show the sampling effectiveness of the MOMCMC method.

The rest of the paper is organized as follows. Section 2 discusses the general MCMC methods when multiple objective functions are present. The implementation of the MOMCMC method is described in detail in Section 3. Section 4 presents the computational results of MOMCMC on the ZDT benchmark functions [20] and other test functions in comparison with NSGA-II and MOSCEM. Finally, Section 5 concludes our summary and future research directions.

2. Monte Carlo Methods in the Presence of Multiple Objective Functions

Suppose that we are interested in exploring solutions **x** that minimize an objective function $f(\mathbf{x}) \in \mathbb{R}$, where $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$. The fundamental formulation of MCMC is to construct a probability distribution:

$$\pi(\mathbf{x}) \propto e^{-f(\mathbf{x})/T},\ T > 0,$$

where T is the simulated temperature. Then, a Markov chain is constructed to sample $\pi(\mathbf{x})$. Typically, the solutions in the vicinity of the global minimum of $f(\mathbf{x})$ are most likely to be drawn in the sampling process. Therefore, MCMC can also be used for optimization purposes.

In multi-objective sampling, when M objective functions, $f_1(\mathbf{x}), \ldots, f_M(\mathbf{x})$, are presented, the probability distribution $\pi(\mathbf{x})$ is converted to

$$\pi(\mathbf{x}) \propto e^{-dom(\mathbf{x})/T},\ T > 0,$$

where $dom(\mathbf{x})$ is the dominance function. Without loss of generality by assuming that all objectives are for minimization, the definition of dominance function is based on the concept of Pareto dominance [16]. A solution $\boldsymbol{u}$ is said to Pareto dominate another solution $\boldsymbol{v}$ ($\boldsymbol{u} \prec \boldsymbol{v}$) if both following conditions i) and ii) are satisfied:

i) for each objective function $f_i(.)$, $f_i(\boldsymbol{u}) \le f_i(\boldsymbol{v})$ holds for all i; and
ii) there is at least one scoring function $f_j(.)$ where $f_j(\boldsymbol{u}) < f_j(\boldsymbol{v})$ is satisfied.

Consequently, the dominance function, $dom(\mathbf{x})$, can be measured as the area containing all possible solutions that dominate **x**. FIG. 2.1 shows the schematic illustration of the dominance function values of a Pareto optimal solution (A) and a non-Pareto optimal solution (B) in the function space of two objective functions f_1 and f_2.

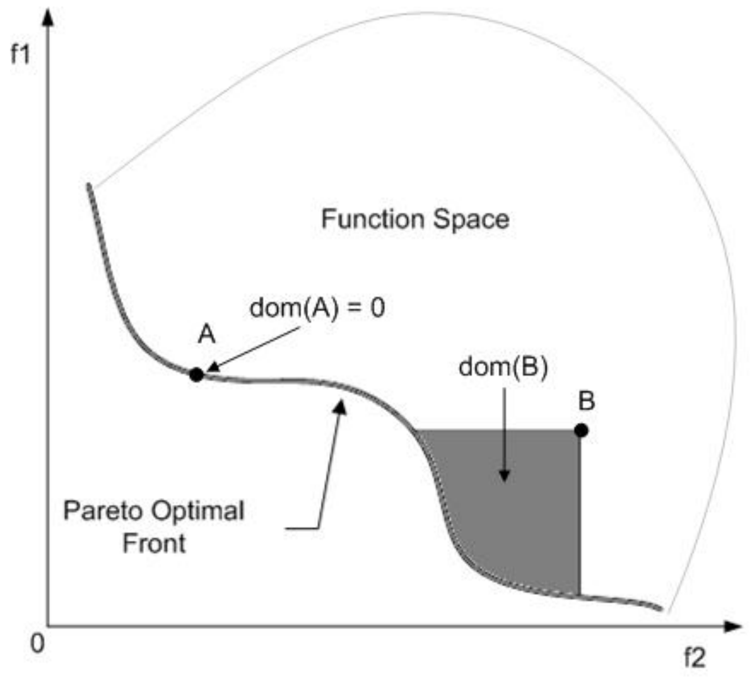


FIG. 2.1. Dominance function in the function space of two objective functions f_1 and f_2. A is a Pareto optimal solution, with *dom*(A) = 0.0. B is not at the Pareto optimal front, where the dominance function value *dom*(B) is measured by the grey area shown in the figure.

As a result, MCMC for multi-objective sampling can be formulated as a sampling problem of the dominance function, where a collection of Pareto optimal solutions corresponds to the global minimum of the dominance function with value 0 while the near Pareto optimal solutions yield relatively small $dom(\mathbf{x})$ values. Unfortunately, in practice, since the Pareto optimal front is usually unknown, it is very difficult to explicitly calculate the exact dominance function value for a solution. Instead, a population of solution samples is created and the Pareto dominance relationship is evaluated between each pair of the solutions, which can be used to estimate the dominance function value for each solution. By sampling and minimizing the dominance function, the MCMC sampler evolves the population of Markov chains toward the solutions at the Pareto optimal front in the objective function space.

3. Multi-Objective Markov Chain Monte Carlo (MOMCMC)

MOMCMC is a population-based Markov Chain Monte Carlo method. Differential Evolution (DE) is used to propose new solutions in MCMC transition. A fitness assignment scheme balancing optimality and diversity is adopted in MOMCMC while dominance significance is taken into consideration. The MCMC acceptance rate is used to control the solution sampling bandwidth. Traditional convergence analysis based on autocorrelation can also be applied to analyze convergence in MOMCMC.

3.1. Population-based MCMC

MOMCMC is a population-based method where the population P includes N individual solutions, $\boldsymbol{c}_1$, …, $\boldsymbol{c}_N$. Correspondingly, N parallel Markov chains are constructed to evolve whilst allowing genetic algorithm style crossover among solutions. The main advantage of the population-based MCMC for multi-objective sampling is that the population, P, simultaneously represents various characteristics of the solution space, which can be used to estimate the dominance function based on the dominance relationship among the solutions in *P*.

3.2. Differential Evolution

Differential Evolution (DE) [14] is a genetic algorithm-based global optimization method popularly used in a variety of scientific and engineering applications for continuous function optimization. Because of its fast convergence, Ter Braak [17] has incorporated DE into the MCMC scheme to improve the sampling capability of a target distribution in the random walk Metropolis method [17].

In MOMCMC, DE is used to propose new solutions based on the current population. In the DE scheme [14], for each solution, $\boldsymbol{c}_i = (x_1, \ldots, x_n)$, a trial parameter vector $\boldsymbol{v}$ is typically formed by

$$\boldsymbol{v} = \boldsymbol{c}_{r1} + F(\boldsymbol{c}_{r2} - \boldsymbol{c}_{r3}),$$

where *r1*, *r2*, and *r3* are mutually distinct, different from *i*, and uniformly distributed integer random numbers in the interval [1, *N*], and $F > 0$ is a tunable amplification control constant as described in [14]. Other forms of trial parameter vector generation are described in [14], which can also be employed in MOMCMC as well. Then, a proposed new solution $\boldsymbol{c}_i' = (x_1', \ldots, x_n')$ is generated by the crossover operation on $\boldsymbol{v}$ and $\boldsymbol{c}_i$:

$$x_j' = \begin{cases} v_j & j = \langle k \rangle_n, \langle k+1 \rangle_n, \ldots, \langle k+L \rangle_n \\ x_j & otherwise \end{cases},$$

where $\langle . \rangle_n$ denotes the modulo operation with modulus *n*, *k* is a randomly generated integer from the interval [0, *n*-1], *L* is an integer drawn from [0, *n*-1] with probability $\Pr(L = l) = (\mathrm{CR})^l$, and $\mathrm{CR} \in [0, 1)$ is the crossover probability. Practical advice suggests that CR = 0.9 and F = 0.8 are suitable choices in the DE scheme [14]. These parameter values are also adopted in our MOMCMC method. The acceptance of new solution $\boldsymbol{c}_i'$ into new population *P'* will be determined later by the Metropolis-Hastings rule [6, 7].

The fundamental idea behind DE is to generate trial parameter vectors by crossing over several randomly selected solutions within the current population. The trial parameter vectors represent random deviations based on the distance and direction information extracted from the population, which can be used to form an adaptive proposal function to guide the generation of new solutions in MOMCMC. Compared to the proposal function used in MOSCEM [12], generating new solutions using DE style crossover no longer relies on predefined probability distribution functions. Moreover, the DE-based proposal function takes advantage of the distance and direction information in the current population, which leads to more precise jumps and thus results in faster MCMC convergence compared to MOSCEM's random walk-based proposal function.

3.3. Fitness Assignment

Multi-objective functions sampling has two equally important goals: 1) optimality – finding the Pareto optimal and near Pareto optimal solutions; and 2) diversity – obtaining diversified coverage of the Pareto optimal front. Because the dominance function does not take diversity into consideration, sampling the dominance function may result in clustering of solutions in certain regions of the objectives while missing the other near Pareto optimal solutions. To preserve diversity in sampling, a fitness function is designed in MOMCMC to replace the dominance function by giving preference to the undersampled solutions. Consequently, the MOMCMC sampler evolves based on the fitness function instead of the dominance function.

MOSCEM [12] uses a fitness assignment which is based on to the number of external non-dominated points. Considering a population *P* with *N* individual solutions, $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_N$, the MOSCEM fitness calculation is based on the strength $s(\boldsymbol{c}_i)$ of each solution $\boldsymbol{c}_i$. The strength $s(\boldsymbol{c}_i)$ is defined as the proportion of solutions in *P* dominated by $\boldsymbol{c}_i$. Then, the fitness of an individual solution $\boldsymbol{c}_i$ is defined as

$$fit_{MOSCEM}(\boldsymbol{c}_i) = \begin{cases} s(\boldsymbol{c}_i) & \boldsymbol{c}_i \text{ is non-dominated} \\ 1 + \sum_{\mathrm{j}}^{\boldsymbol{c}_{\mathrm{i}} \succ \boldsymbol{c}_{\mathrm{j}}} s(\boldsymbol{c}_j) & \boldsymbol{c}_i \text{ is dominated} \end{cases} .$$

Fitness value of 1.0 separates the dominated and non-dominated solutions in a population – the solutions with fitness less than 1.0 are non-dominated while the others are dominated. The fitness function biases to the non-dominated solutions with less dominated ones while those with a lot of neighbors in their niche are penalized [12].

The fitness assignment proposed in MOSCEM nicely balances optimality and diversity. However, the key disadvantage of this fitness assignment scheme is that the calculation of the strength of a dominated solution is not able to differentiate the significance of dominance. Consider the example shown in FIG. 3.1 with three solutions A, B, and C under two objective functions *f1* and *f2*, where B and C are both dominated by A. According to the MOSCEM fitness assignment scheme, the MOSCEM fitness values of A, B, and C are 0.67, 1.67, and 1.67, respectively, since B and C are not dominating other solutions. As a result, the fitness function cannot differentiate the favorability between B and C, although compared to B, C seems to be more "significantly" dominated by A and should be less favorable.

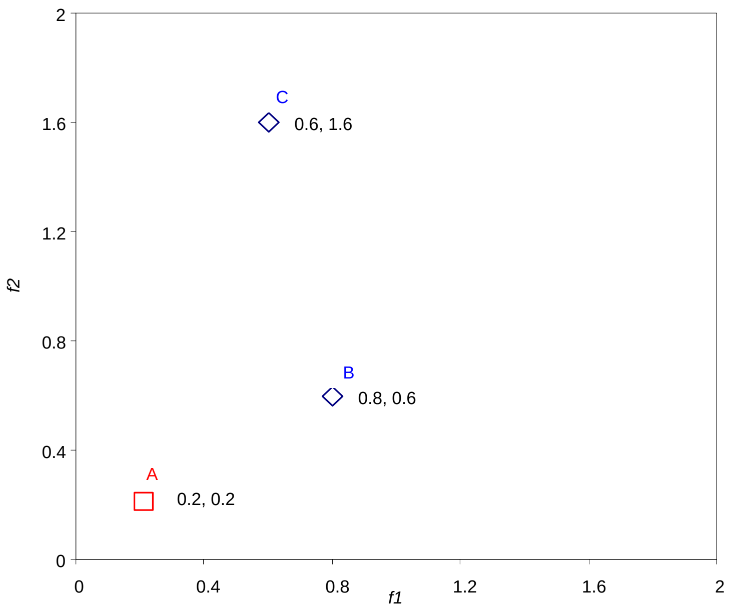


FIG. 3.1. A schematic example for comparison of the fitness functions in MOSCEM and MOMCMC. The MOSCEM fitness function cannot differentiate the two dominated solutions B and C (fit_{MOSCEM}(B) = fit_{MOSCEM}(C) = 1.67) by simply counting the number of solutions B or C dominating. By measuring the pair-wise hypervolumes between B, C and the non-dominated solution A, solution B (fit_{MOMCMC}(B) = 1.24) is more favorable than C (fit_{MOMCMC}(C) = 1.56) in the MOMCMC fitness function.

The MOMCMC method improves the fitness assignment scheme in MOSCEM by measuring the solution pair-wise hypervolume to estimate the significance of dominance. If a solution $\boldsymbol{c}_i$ is dominated by another solution $\boldsymbol{c}_j$, i.e., $\boldsymbol{c}_j \prec \boldsymbol{c}_i$, the significance of the domination can be measured by the pair-wise hypervolume between $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$, $HYP(\boldsymbol{c}_i, \boldsymbol{c}_j)$, which is defined as

$$HYP(\boldsymbol{c}_i, \boldsymbol{c}_j) = VOL(\boldsymbol{c}_i, \boldsymbol{c}_j),$$

where $VOL(.)$ is the Lebesgue measure. Then, the fitness function in MOMCMC is defined as

$$fit_{MOMCMC}(\boldsymbol{c}_i) = \begin{cases} s(\boldsymbol{c}_i) & \boldsymbol{c}_i \text{ is non-dominated} \\ 1 + \sum_{j}^{c_j \prec c_i} HYP(\boldsymbol{c}_i, \boldsymbol{c}_j) & \boldsymbol{c}_i \text{ is dominated} \end{cases}.$$

For the non-dominated solutions, we keep the same fitness definition as MOSCEM so that those solutions dominating less other solutions are biased so as to favor diversity. For each dominated solution, we calculate the significance of domination by accumulating the hypervolumes between it and the solutions dominating it. For the example shown in FIG. 3.1, fit_{MOMCMC}(B) is 1.24 and fit_{MOMCMC}(C) is 1.56, where the favorability between B and C can be distinguished by the MOMCMC fitness function.

3.4. Markov Chain Monte Carlo

The Metropolis-Hastings acceptance rule is employed to determine acceptance of the proposed solution generated by DE in MOMCMC. The transition probability depends on the difference of the fitness function values. For each proposed solution $\boldsymbol{c}_i'$ generated from $\boldsymbol{c}_i$, the Metropolis-Hastings ratio is calculated by:

$$w(\boldsymbol{c}_i \rightarrow \boldsymbol{c}_i') = e^{\frac{-(fit_{MOMCMC}(\boldsymbol{c}_i') - fit_{MOMCMC}(\boldsymbol{c}_i))}{T}}.$$

The new solution $\boldsymbol{c}_i'$ is accepted with the probability

$$\min(1, w(\boldsymbol{c}_i \rightarrow \boldsymbol{c}_i')).$$

T is the simulated temperature, which can be used to control the acceptance rate of the Metropolis-Hastings transitions.

Since in the DE scheme, the three random integers, *r1*, *r2*, and *r3*, used in producing the trial parameter $\boldsymbol{v}$ are uniformly distributed, transition from $\boldsymbol{c}_i$ to $\boldsymbol{c}_i'$ and the reverse transition have the same probability in producing $\boldsymbol{v}$, i.e., the transition probability from $\boldsymbol{c}_i$ to $\boldsymbol{c}_i'$ is equal to that of its reverse transition. With acceptance probability $\min(1, w(\boldsymbol{c}_i \rightarrow \boldsymbol{c}_i'))$, the DE transitions in MOMCMC satisfy detailed balance. A rigorous proof can be found in [17]. Therefore, the resulting Markov chain is reversible and thus the MOMCMC sampler can lead to a stationary distribution, where the global minimum corresponds to the population of solutions diversely and uniformly covering the Pareto optimal front of the function space composed of multiple objective functions.

3.5. Acceptance Rate

As mentioned in section 1, in multi-objective sampling, we are not only interested in finding the solutions at the Pareto optimal front, but also those close to the Pareto optimal front with diversified configurations. The solution bandwidth near the Pareto optimal front is controlled by the acceptance rate in MOMCMC, which is maintained by adjusting the simulated temperature T during the sampling process. Higher acceptance rates allow the MOMCMC sampler a higher chance of exploring solutions far away from the Pareto optimal front whilst a lower acceptance rate more likely prevents MOMCMC from further deviating from the Pareto optimal front.

3.6. Convergence Estimation

An important question is when MOMCMC will converge. Similar to the other MCMC methods, one can evaluate the autocorrelation function in MOMCMC to estimate the number of iterations to reach equilibrium (convergence). Given some "observable" O_t at iteration time t in the Markov process, the autocorrelation function is defined as

$$\Gamma(T) = \frac{E[(O_t - \mu)(O_{t+T} - \mu)]}{\sigma^2} ,$$

where μ is the mean and σ^2 is the variance. Then, the Integrated Autocorrelation Time (IAT) [18] can be evaluated as

$$\tau \cong \tfrac{1}{2} + \sum_{T=1}^{\infty} \Gamma(T) .$$

Studies [19] in MCMC suggest that the number of iterations, *NUM*, for the MCMC sampler to settle in equilibrium should be, *NUM* >> $\boldsymbol{\tau}$.

3.7. MOMCMC Pseudocode

By putting all the pieces together, the MOMCMC method is described as the pseudocode shown in FIG. 3.2. It is important to notice that, similar to MOSCEM, the MOMCMC method is particularly suitable for parallel implementation because the computations on the new solutions, including generation, objective functions evaluation, fitness assignment, and acceptance determination, can be carried out independently.

```
[Initialization]
    Randomly produce population $P$ of $\boldsymbol{c}_1,\dots,\boldsymbol{c}_N$
    Initialize temperature $T$
[MOMCMC]
    Repeat {
      Calculate domination count of each solution $\boldsymbol{c}_i$ in $P$
      Evaluate $fit_{MOMCMC}(\boldsymbol{c}_1),\dots,fit_{MOMCMC}(\boldsymbol{c}_N)$
      $P' = \phi$                  // New population
      for $i$ = 1 to $N$ {
         [Differential Evolution]
         Generate $\boldsymbol{c}_i'$ for $\boldsymbol{c}_i$ using DE crossover
         [Fitness Assignment]
         Evaluate dominance relationship between $\boldsymbol{c}_i'$ and $\boldsymbol{c}_1,\dots,\boldsymbol{c}_N$
         Evaluate $fit_{MOMCMC}(\boldsymbol{c}_i')$ against $P$
         [Markov Chain Monte Carlo]
         $\theta = U[0, 1)$        // Produce a uniform random number
         if $(\theta < \exp(-(fit_{MOMCMC}(\boldsymbol{c}_i')-fit_{MOMCMC}(\boldsymbol{c}_i))/T))$
            $\boldsymbol{P'} = \boldsymbol{P'} \cup c_i'$     // Accept
         else
            $\boldsymbol{P'} = \boldsymbol{P'} \cup c_i$      // Reject
      }
      $\boldsymbol{P} = \boldsymbol{P'}$                  // Replace old population
    [Adjust temperature]
    Compute acceptance rate $\theta$
    if $\theta < \theta_{desired}$
      increase $T$
    else if $\theta > \theta_{desired}$
      decrease $T$
    } Until convergence is reached and sufficient number of
    samples are obtained
```

FIG. 3.2. Pseudocode of the MOMCMC Method

4. Computational Results

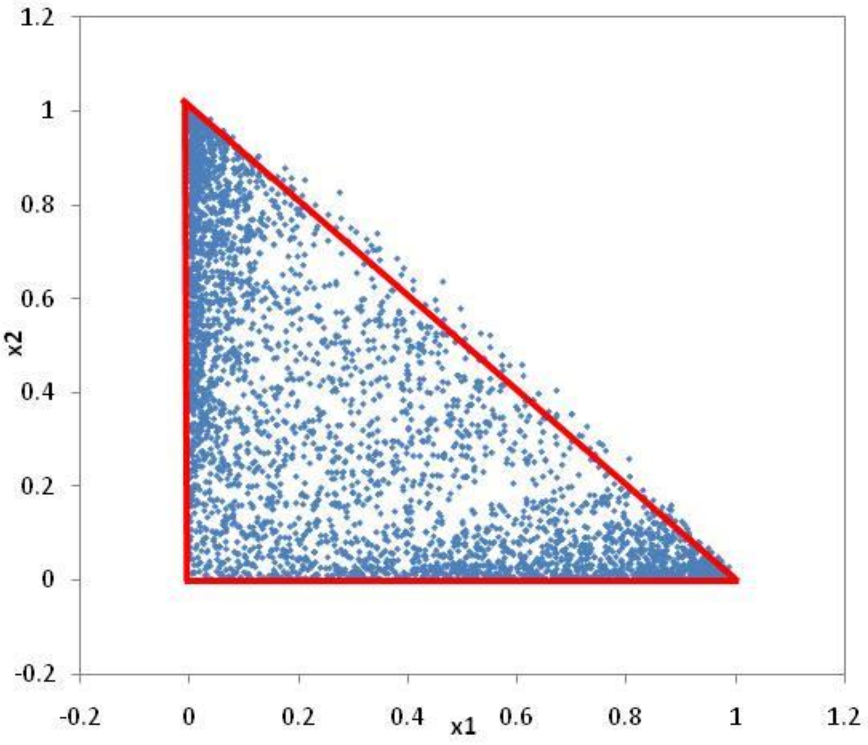
We apply the MOMCMC method to a variety of multi-objective benchmark problems in real parameter space. In subsection 4.1, the MOMCMC sampling results on a simple multi-objective problem with three objective functions are compared with those generated by multi-objective optimization using NGSA-II. We also analyze the performance of MOMCMC and compare the speed of convergence between MOMCMC and MOSCEM on ZDT benchmark functions [20] in subsections 4.2 and 4.3, respectively. Furthermore, in subsection 4.4, we use MOMCMC with relatively small population size to sample the modified ZDT3 functions, which have more separated Pareto optimal front segments than population size, to show its effectiveness on complex multi-objective problems with complicated function space. The computational experiments were carried out on a server with an Intel i7 CPU 920 @ 2.67GHz, 8MB cache, and 6G memory. Our program implementation of the MOMCMC is written in C++ and is compiled by GNU g++ compiler.

4.1. Multi-Objective Sampling vs. Multi-Objective Optimization

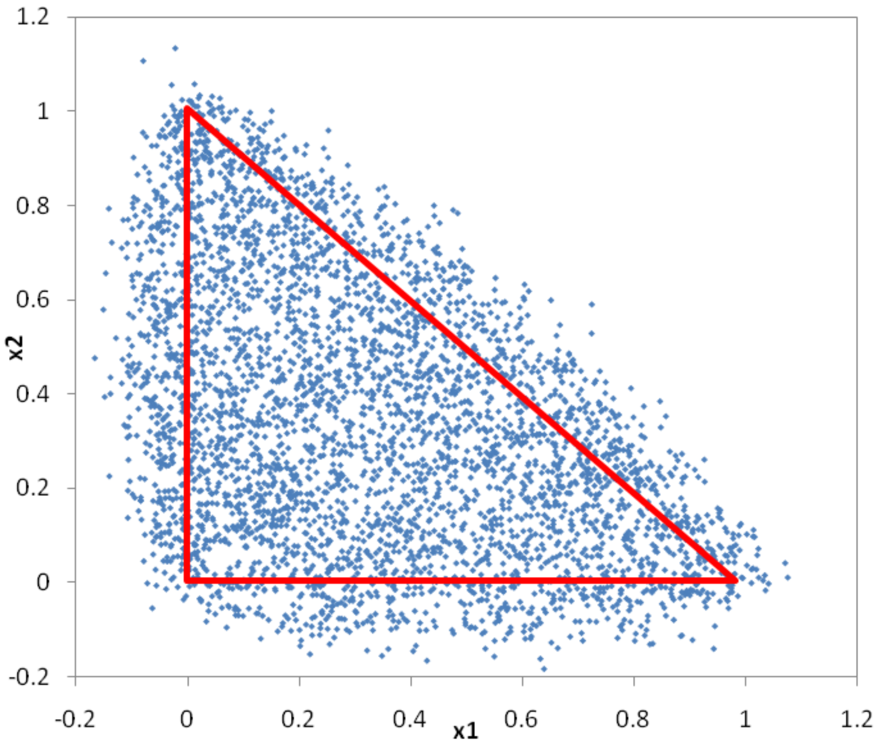
We apply the MOMCMC method to a simple multi-objective problem with two variables and three objective functions, which is used to show the effectiveness of MOSCEM in [12]. The three objective functions in this simple multi-objective problem are defined as:

$$f_1(x_1, x_2) = x_1^{\ 2} + x_2^{\ 2}$$
$$f_2(x_1, x_2) = (x_1 - 1)^2 + x_2^{\ 2}$$
$$f_3(x_1, x_2) = x_1^{\ 2} + (x_2 - 1)^2 ,$$

where the Pareto optimal solutions lay in the triangular-shaped area with the corner points (0, 0), (0, 1), and (1, 0) in the solution space. FIG. 4.1(a) shows the scatter plot of the population of 4,096 solutions in NSGA-II after 100 iterations, where most of these solutions are located in the Pareto optimal front. This is because NSGA-II is an optimization algorithm, which tries to eliminate all the non-dominated solutions during the optimization process. In contrast, FIG. 4.1(b) shows a snapshot of a population sample with the 4,096 solutions in MOMCMC after 100 iterations with 15% acceptance rate. MOMCMC has demonstrated uniform coverage of the Pareto optimal solutions in this simple multi-objective problem, which is similar to that of MOSCEM. Compared to NSGA-II, MOMCMC also explores solutions near the Pareto optimal front, as shown in FIG. 4.1(b). MOMCMC serves on the sampling purpose, which is interested in sampling the Pareto optimal solutions as well as those near Pareto optimal. The closer the solutions are to the Pareto optimal front, the higher chance they are to be sampled and thus have higher density. This is due to the fact that, for sampling purpose, MOMCMC not only accepts good moves with better fitness, but also accepts bad moves leading to non-dominated solutions with certain probability. The acceptance probability for solutions near the Pareto optimal front is higher than those far away deviated. It is also interesting to notice that the 4,096 solutions found in NSGA-II are clustered in the two acute angles of the triangle-shape Pareto optimal front while the area near the hypotenuse is undersampled. Compared with NSGA-II, MOMCMC leads to a more uniform coverage of the Pareto optimal front in the solution space. In many applications such as drug design [4], protein structure modeling [5, 26], and circuit design [27], finding a diverse range of solutions close to the Pareto optimal front is typically concerned. The uniform coverage of the Pareto optimal front in the solution space can lead to highly diversified solutions for these applications.



(a)



(b)

FIG. 4.1. Comparison between NSGA-II and MOMCMC in a simple 2D multi-objective problem. NSGA-II serves the optimization purpose where most solutions obtained are at the Pareto optimal front. MOMCMC serves the sampling purpose, which is not only interested in sampling solutions at the Pareto optimal front, but also those near the front. MOMCMC also covers the Pareto optimal front in the solution space more uniformly than NSGA-II. (a) Population of 4,096 solutions generated by NSGA-II after 100 iterations. (b) Population of 4,096 solutions generated by MOMCMC after 100 iterations.

4.2. ZDT Multi-Objective Problems

We test the MOMCMC method on a set of benchmark functions [20] with different Pareto optimal front characteristics, which has been used broadly in the multi-objective optimization literature. These test functions include ZDT1 (convex Pareto optimal front), ZDT2 (nonconvex Pareto optimal front), ZDT3 (Pareto optimal front with several noncontiguous convex parts), ZDT4 (Pareto optimal front with multimodality), and ZDT6 (nonuniform search space)[1], which have been used to study MOSCEM in recent work [13]. MOSCEM has been found to have unsatisfactory performance on these test functions which include poor convergence to the Pareto optimal front as well as using a significant amount of computational time.

We measure the dominated volume of the non-dominated solutions using the hypervolume indicator [21]. Considering a set of solutions *S* with *N* solutions, $\boldsymbol{c}_1$, …, $\boldsymbol{c}_N$, the hypervolume indicator, *HYP(S)*, relative to an arbitrarily selected reference point $\boldsymbol{r}$, is defined as

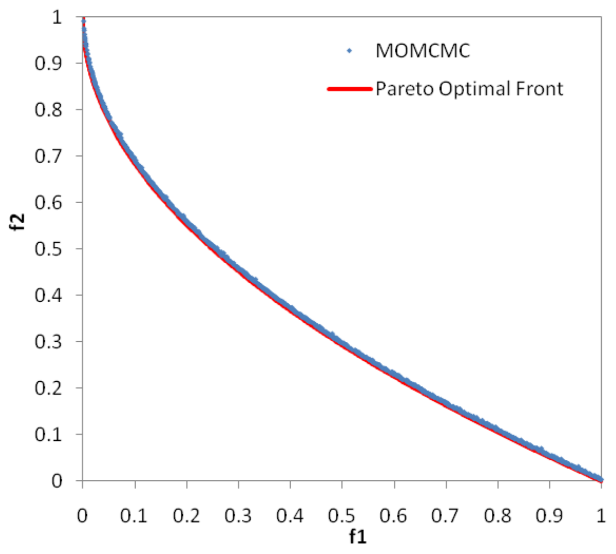
$$HYP(S) = \bigcup_i VOL(\boldsymbol{f}(c_i), r),$$

where *VOL*(.) is the Lebesgue measure and $\boldsymbol{f}(\boldsymbol{c}_i)$ is the corresponding point of $\boldsymbol{c}_i$ in the objective function space. Based on the definition of the hypervolume indicator, the hypervolume value is solely determined by the non-dominated solutions and the reference point while the dominated solutions do not contribute. Typically, a higher hypervolume value indicates that the set of solutions *S* is closer to the Pareto optimal front and/or more diversified.

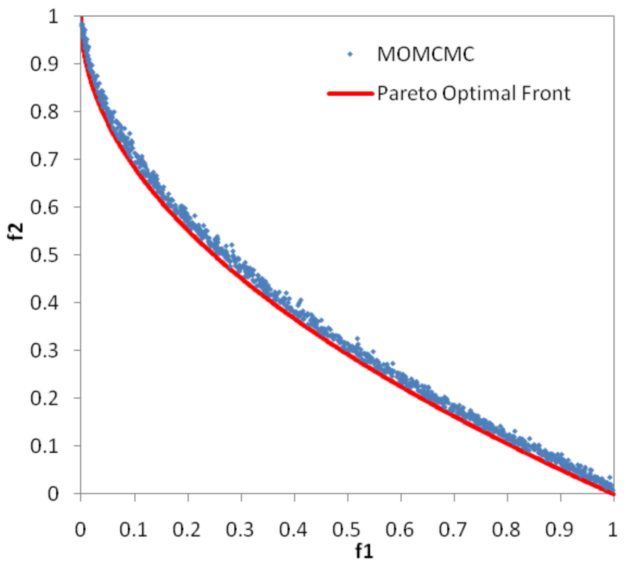
FIG. 4.2-4.6 illustrate the computational results of MOMCMC on 100 dimensional versions of ZDT1-4 and ZDT6. The computation adopts the suggested DE parameters (CR = 0.9 and F = 0.8), a 15% acceptance rate in MCMC and uses a population of 1,024. The reference point is at (1.0, …, 1.0). Figures (a) in FIG. 4.2-4.6 show the sets of solutions with the maximum hypervolume values found during MOMCMC sampling, which exhibit diversified coverage of the Pareto optimal fronts in all ZDT benchmark functions. Figures (b) and (c) in FIG. 4.2-4.6 depict two samples of solutions with smaller hypervolume values obtained in MOMCMC sampling after equilibrium is reached, which show that MOMCMC can also sample solutions near the Pareto optimal front, and the hypervolume values indicate how far the solutions in the sample are deviated from the Pareto optimal front. Table 4.1 summarizes the sampling performance of MOMCMC on the ZDT benchmark functions.

The samples obtained in MOMCMC also reveal the characteristics of the function space formed by the multiple objective functions. For example, in ZDT6, unlike the other ZDT test functions, solutions in MOMCMC samples with lower hypervolume values are mostly deviated from the Pareto optimal front. This is due to the fact that the distribution of the solutions in ZDT6 functions are non-uniform, where the solution density is lowest at the Pareto optimal front and increases when being away from the front [20].

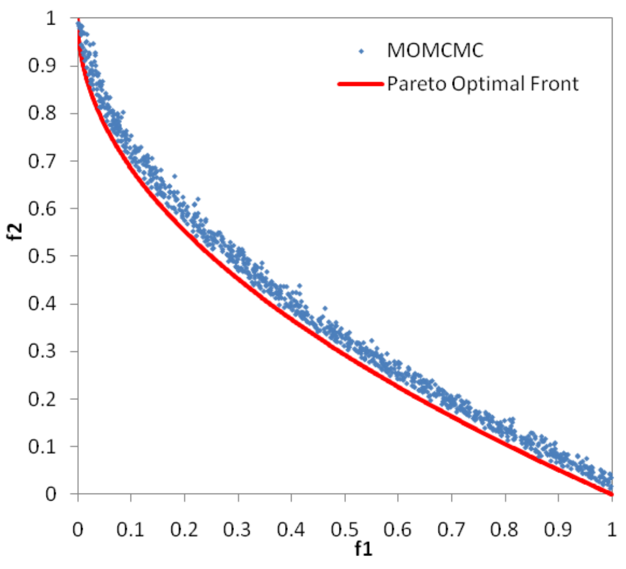
[1] ZDT5 is in discrete parameter space



(a)

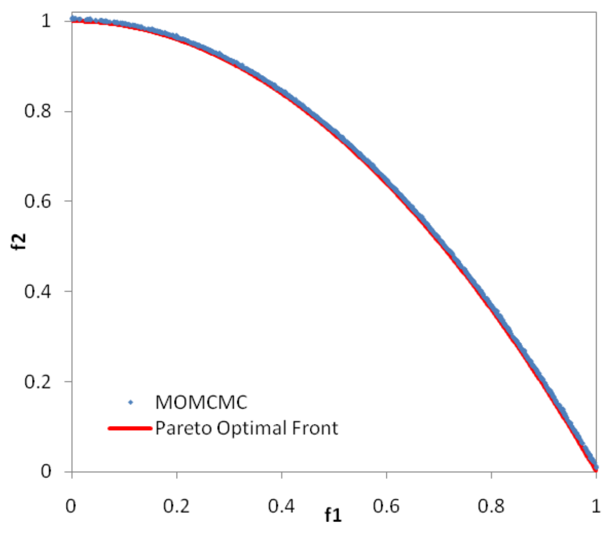


(b)

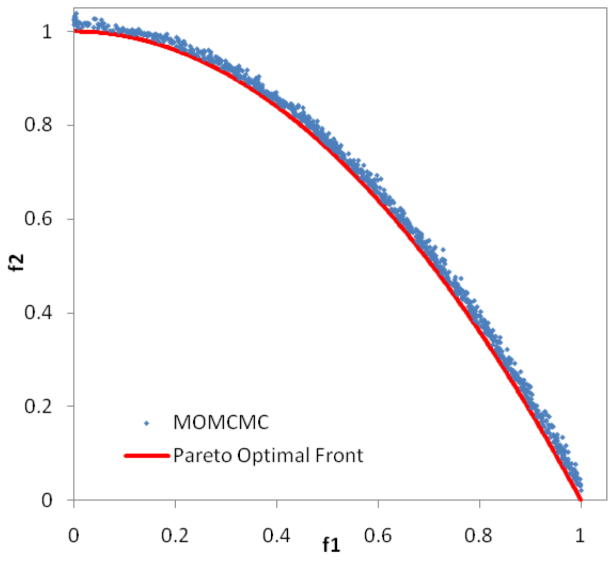


(c)

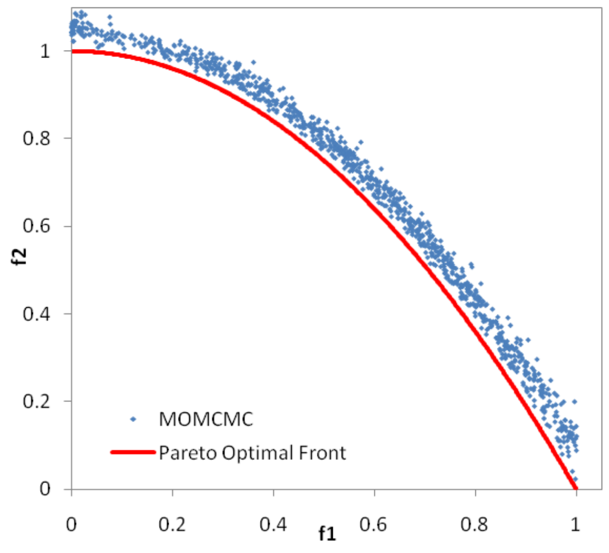
FIG. 4.2. Computational results of MOMCMC on 100 dimensional ZDT1. (a) Solutions in a sample with HYP = 0.662 (largest hypervolume found in sampling). (b) Solutions in a sample with HYP = 0.653. (c) Solutions in a sample with HYP = 0.645



(a)



(b)



(c)

FIG. 4.3. Computational results of MOMCMC on 100 dimensional ZDT2. (a) Solutions in a sample with HYP = 0.328 (largest hypervolume found in sampling). (b) Solutions in a sample with HYP = 0.310. (c) Solutions in a sample with HYP = 0.302.

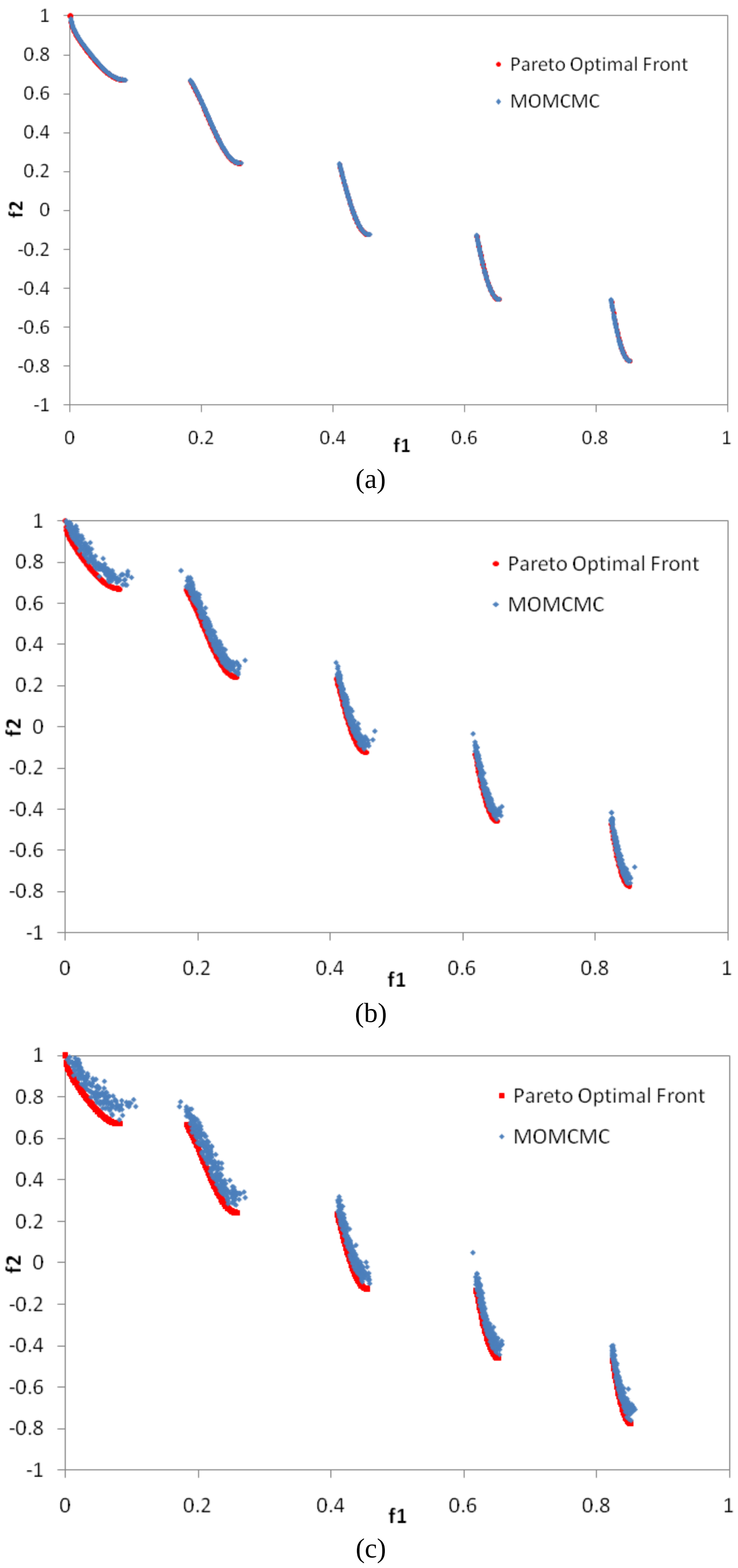
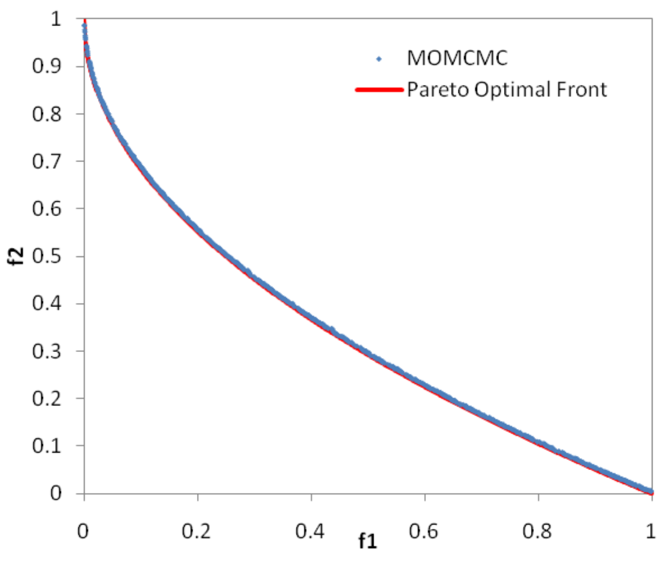
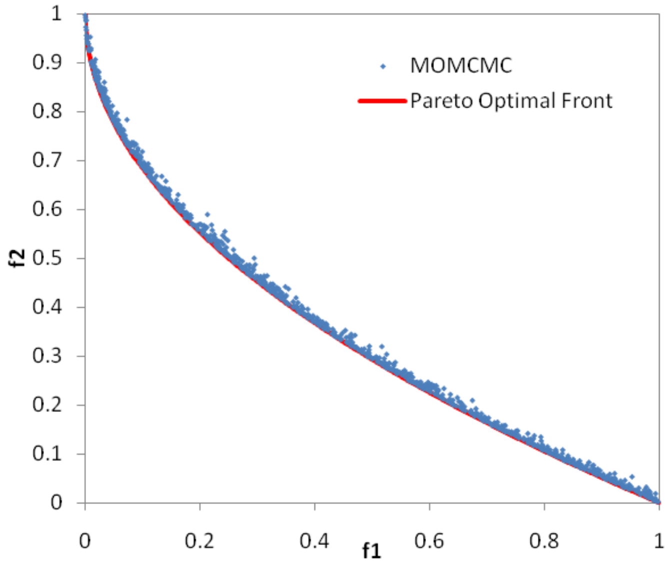


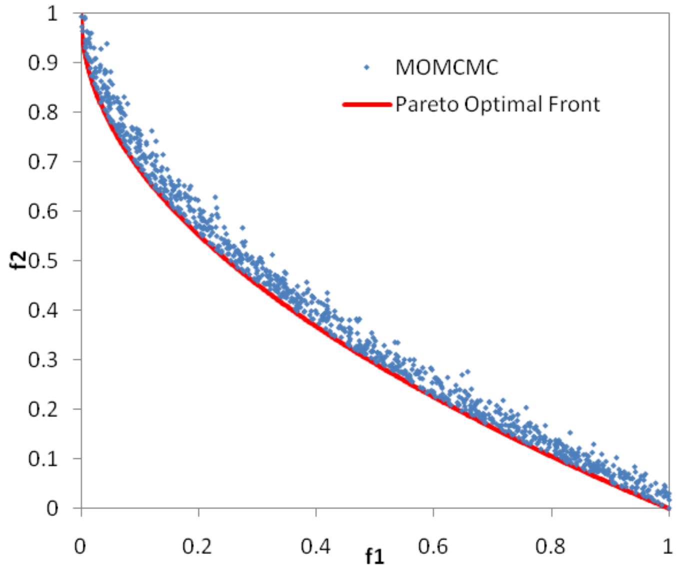
FIG. 4.4. Computational results of MOMCMC on 100 dimensional ZDT3. (a) Solutions in a sample with HYP = 1.042 (largest hypervolume found in sampling). (b) Solutions in a sample with HYP = 1.028. (c) Solutions in a sample with HYP = 1.017.



(a)

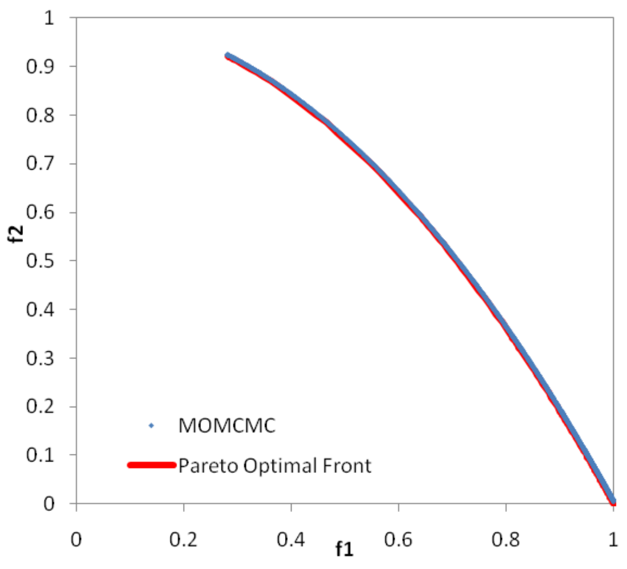


(b)

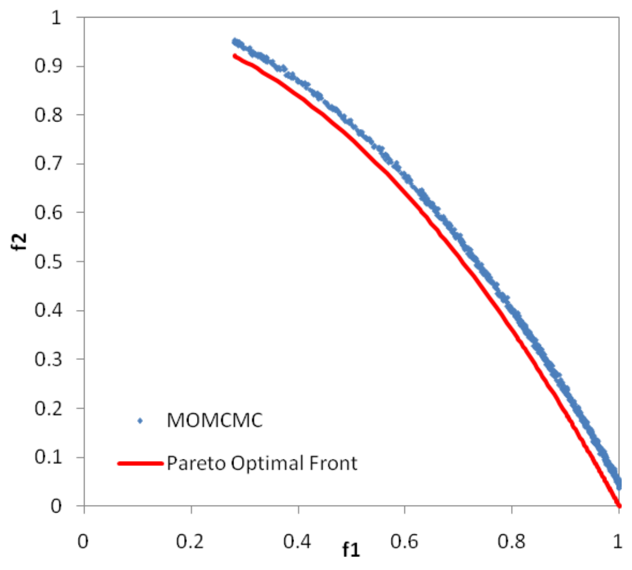


(c)

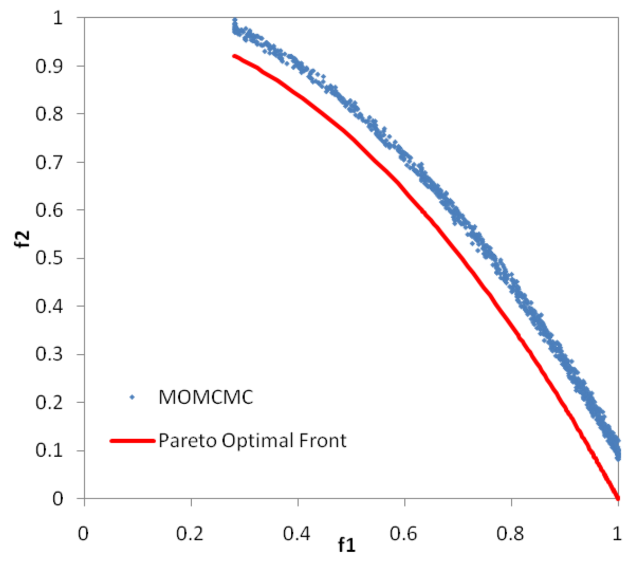
FIG. 4.5. Computational results of MOMCMC on 100 dimensional ZDT4. (a) Solutions in a sample with HYP = 0.665 (largest hypervolume found in sampling). (b) Solutions in a sample with HYP = 0.661. (c) Solutions in a sample with HYP = 0.656



(a)



(b)



(c)

FIG. 4.6. Computational results of MOMCMC on 100 dimensional ZDT6. (a) Solutions in a sample with HYP = 0.316 (largest hypervolume found in sampling). (b) Solutions in a sample with HYP = 0.303. (c) Solutions in a sample with HYP = 0.279

Table 4.1. Summary of sampling performance of MOMCMC on ZDT benchmark functions with population size of 1,024. Reference point is at (1.0, …, 1.0).

	Max HYP	Mean HYP	Standard Deviation
ZDT1	0.662	0.653	0.003
ZDT2	0.328	0.310	0.008
ZDT3	1.042	1.027	0.010
ZDT4	0.665	0.660	0.002
ZDT6	0.316	0.300	0.009

4.3. Convergence Comparison between MOMCMC and MOSCEM

FIG. 4.7 compares the 100-lag autocorrelation plot of the sample time series produced by MOMCMC and MOSCEM in 100 dimensional ZDT6. The autocorrelation indicates the “stickiness” of the MCMC sampler in local modes [22]. One can observe that, in the 100-lag plot, the autocorrelation function value in MOMCMC decreases to less than 0.1 while that in MOSCEM is still as high as about 0.6. One reason is the DE-based proposal scheme produces more adaptive jumps by taking advantage of current population’s distance and direction information than that of MOSCEM. The other reason is that the MOMCMC fitness function can differentiate the dominated solutions more precisely by taking dominance significance into consideration. Consequently, MOMCMC yields a better convergence rate (mixing rate) [23] than MOSCEM.

Better convergence in MOMCMC leads to faster exploration of the solutions near the Pareto optimal front than MOSCEM. FIG. 4.8 compares the standard deviations of the best hypervolume values in 10 independent runs using MOMCMC and MOSCEM in 100 dimensional ZDT1. MOMCMC approaches the optimal hypervolume value (0.6667 – 1,024 points uniformly distributed on the Pareto optimal front with reference point at (1.0, 1.0)) in much less iterations than MOSCEM, which indicates significantly more efficient sampling capability in MOMCMC than MOSCEM. FIG. 4.9 shows that MOMCMC also outperforms MOSCEM in computational time. The main reason is MOMCMC can avoid the costly covariance calculation in MOSCEM.

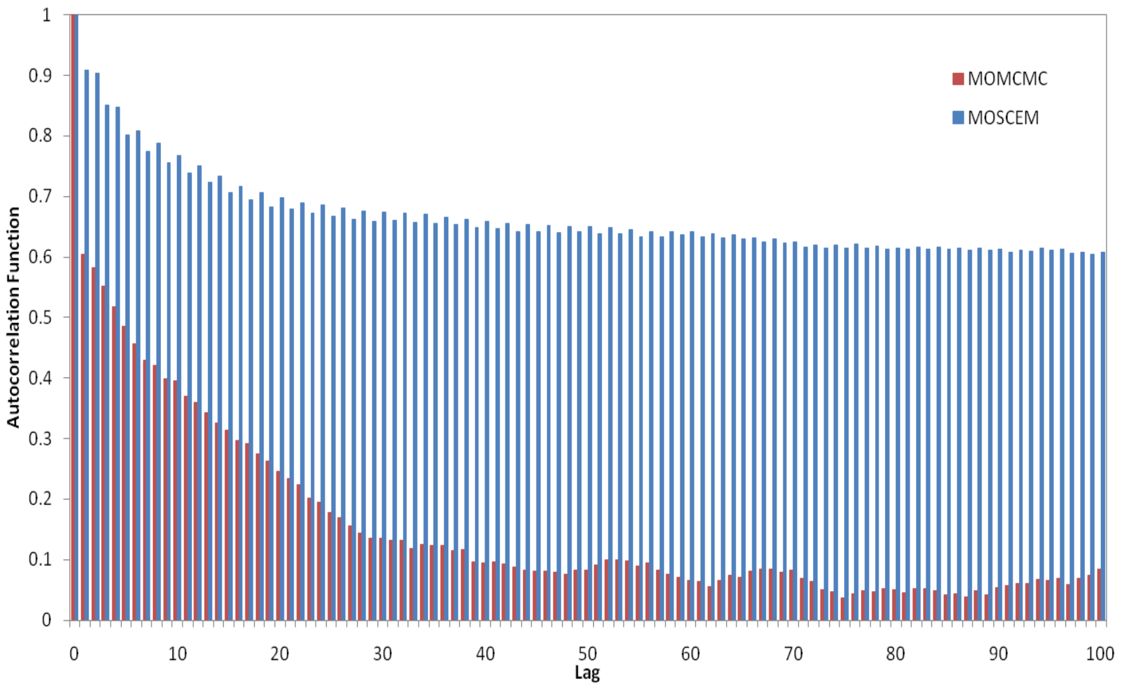


FIG. 4.7. Comparison of autocorrelation functions between MOSCEM and MOMCMC in 100-dimensional ZDT6

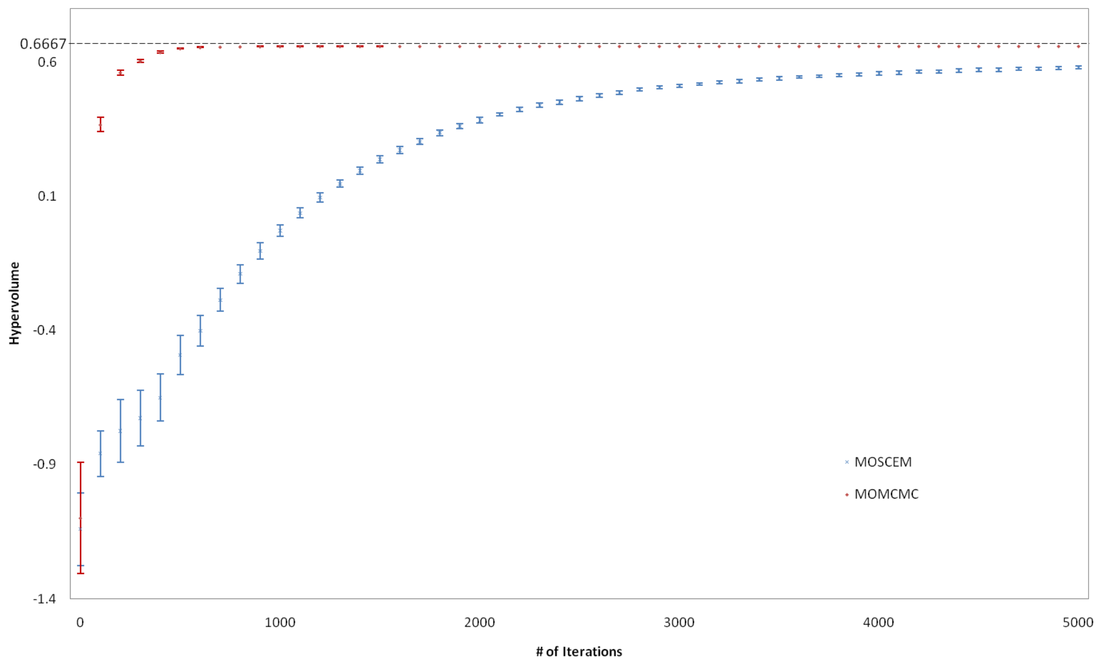


FIG. 4.8. Comparison of the best hypervolume values in 10 runs using MOMCMC and MOSCEM in 100-dimensional ZDT1

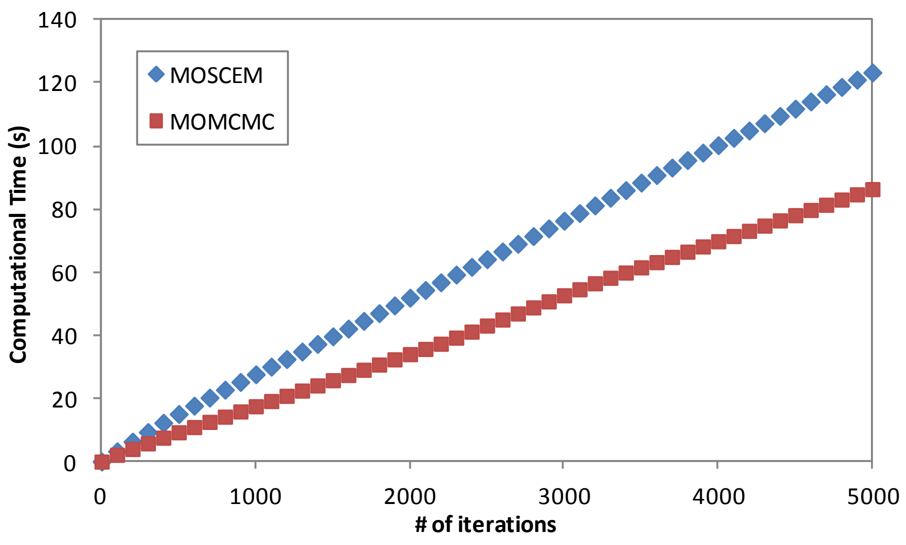


FIG. 4.9. Comparison of computational times of MOMCMC and MOSCEM in 100-dimensional ZDT1. Both MOMCMC and MOSCEM use a population size of 1,024.

4.4. Sampling on Complicated Function Space using Small Population Size

In real-life applications, multiple objective functions can form complex function space with complicated Pareto optimal front shape. The population size used in the multi-objective sampling/optimization algorithm is a critical parameter. Usually, a large population size is required in population-based algorithms in order to sufficiently cover the Pareto optimal front in a complicated multi-objective optimization problem. Unfortunately, employing a large population in sampling is rather costly. For example, the computation complexity of evaluating domination count requires O(MN^2) comparisons in worse case with population size of N and M objective functions [16]. One of the key advantages of multi-objective sampling is to enable one to use a relatively small population size to sample the complex function space and obtain multiple independent samples of solutions to obtain good coverage of the complicated Pareto optimal front.

We consider the following modified ZDT3 functions

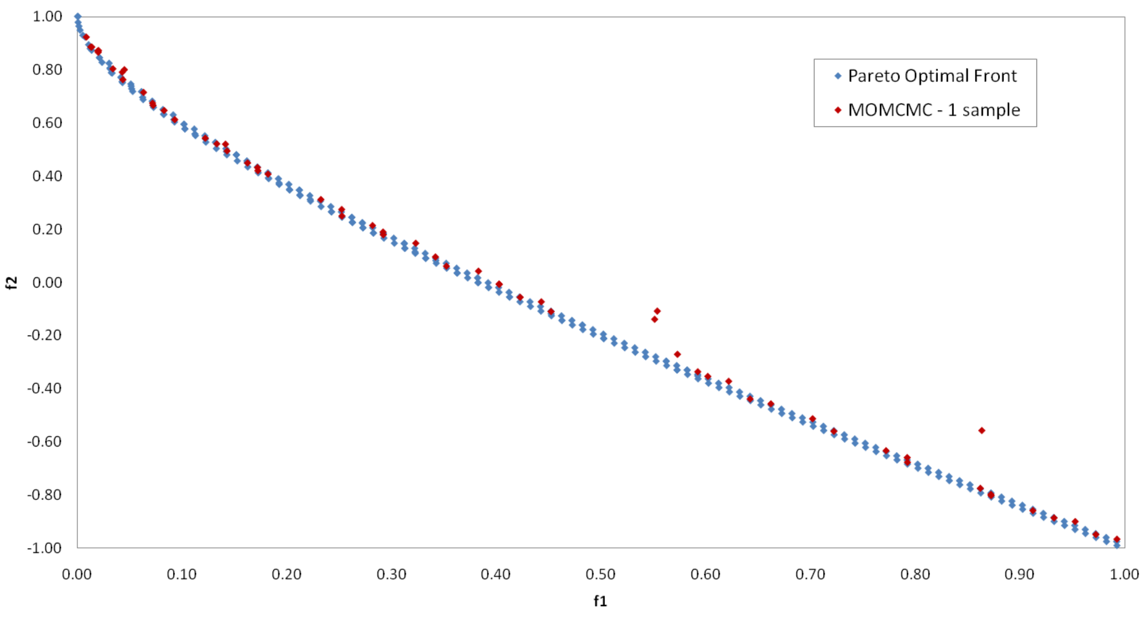
$$f_1(x_1,\ldots,x_N) = x_1$$

$$g(x_2,\ldots,x_N) = 1 + 9 \cdot \sum_{i=2}^{N} \frac{x_i}{N-1}$$

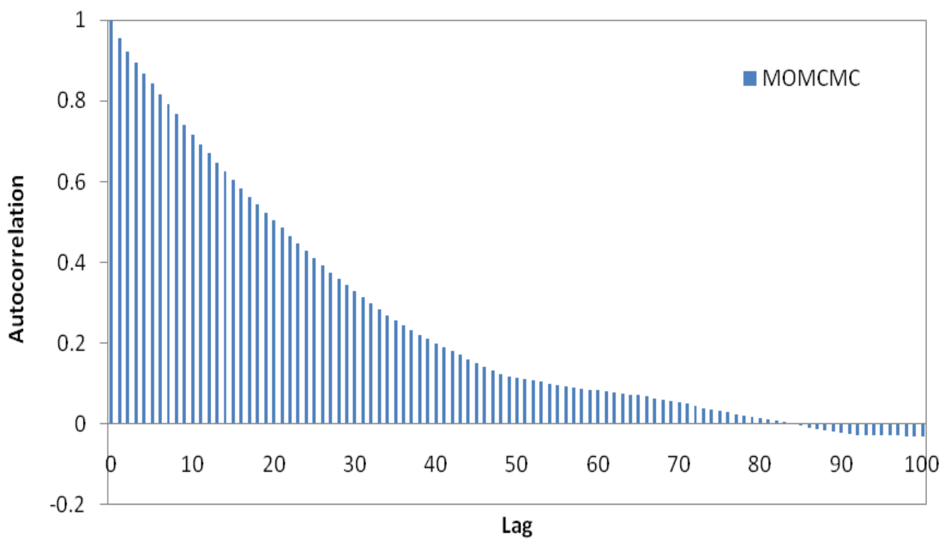
$$h(f_1, g) = 1 - \sqrt{f_1/g} - (f_1/g)\sin(200\pi f_1),$$

$$f_2(x_1,\ldots,x_N) = g \cdot h(f_1, g)$$

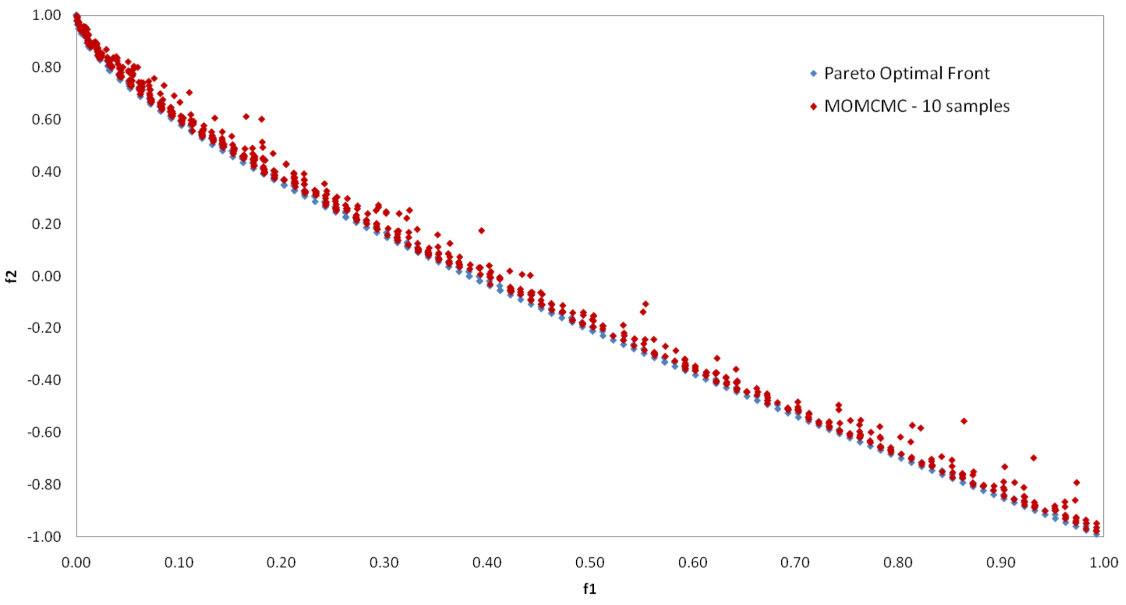
whose Pareto optimal front contains 100 separated segments. FIG. 4.10(a) shows one sample obtained from MOMCMC with a small population size of 64 after equilibrium is reached. Since the population size is less than the number of separated Pareto optimal front segments, a single sample generated by MOMCMC cannot fully cover all Pareto optimal front segments. The autocorrelation plot in FIG. 4.10(b) shows that the autocorrelation function value decreases to -0.02 at lag 100, suggesting that the samples generated in MCMC every 100 iterations can be deemed as independent samples. FIG. 4.10(c) shows the solutions of 10 samples chosen in every 100 lags, which covers all separated Pareto optimal front segments in the modified ZDT3 functions.



(a)



(b)



(c)

FIG. 4.10. Using MOMCMC with small population size to sample complicated function space. (a) Solutions of a single sample generated by MOMCMC with population size of 64 on the modified ZDT3 functions with 100 separated Pareto optimal front segments. (b) Autocorrelation plot of MOMCMC on modified ZDT3 functions with population size of 64. (c) Solutions of 10 samples chosen in every 100 lags using MOMCMC with population size = 64 on the modified ZDT3 functions, which lead to good coverage of the 100 separated Pareto optimal front segments

5. Conclusions

In this paper, we present the MOMCMC method, a new Monte Carlo method to address the multi-objective sampling problem. MOMCMC is a population-based method incorporating a DE-based proposal function into MCMC and measuring the significance of dominance when estimating fitness of dominated solutions. The acceptance rate is used to control the solution sampling bandwidth near the Pareto optimal front, so that the MOMCMC sampler can not only explore solutions at the Pareto optimal front, but also those close to the front. The MOMCMC method has demonstrated its sampling effectiveness in the ZDT test functions, where MOSCEM has certain deficiency. The MOMCMC method is also capable of using small population size to sample complicated function space.

By using DE to propose new solutions, MOMCMC can avoid costly calculation of parameter covariance of a population and thus is scalable with the size of population without imposing significant computational overhead to multi-objective sampling. Therefore, in many multi-objective sampling applications where the most computationally costly part is calculating the objective functions, the bottleneck lies on objective functions evaluation. Fortunately, in most of these applications, evaluation of the objective function values of members in a population can usually be carried out in parallel. As a result, another potential advantage of MOMCMC is its parallelism. Due to the independence of evolving each individual solution in a population, MOMCMC is particularly suitable for the newly emerging high performance computing architectures, such as multicore and general purpose GPU [24], in large-scale multi-objective sampling applications. Our recent implementation of MOMCMC on GPU [25] has shown a speedup of ~100 by taking advantage of the GPU Single-Instruction-Multiple-Threads (SIMT) architecture.

Acknowledgements

The author acknowledges support from NSF grant 1066471 and ODU 2011 SEECR grant.

References

[1] K. Deb, A. Pratap, S. Agarwal, AND T. Meyarivan, *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II*, IEEE Trans. Evol. Computation, 6 (2002), pp. 182-197.

[2] E. Zitzler, M. Laumanns, AND L. Thiele, *SPEA2: Improving the Strength Pareto Evolutionary Algorithm*, TIK-103, Department of Electrical Engineering, Swiss Federal Institute of Technology, Zurich, Switzerland, 2001.

[3] J. Knowles AND D. Corne, *The Pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization*, Proceedings of the 1999 Congress on Evolutionary Computation, 1999.

[4] D. K. Agrafiotis, *Multiobjective optimization of combinatorial libraries*, IBM J. Res. & Dev., 45(3/4) (2001), pp. 545-566.

[5] Y. Li, I. Rata, AND E. Jakobsson, *Integrating Multiple Scoring Functions to Improve Protein Loop Structure Conformation Space Sampling*, Proceedings of IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, 2010.

[6] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, AND E. Teller, *Equation of State Calculations by Fast Computing Machines*, Journal of Chemical Physics, 21 (1953), pp. 1087-1092.

[7] W. K. Hastings, *Monte Carlo Sampling Methods using Markov Chains and Their Applications*, Biometrika, 57 (1970), pp. 97-109.

[8] S. Kirkpatrick, C. D. Gelatt Jr., AND M. P. Vecchi, *Optimization by Simulated Annealing*, Science, 220 (1983), pp. 671-680.

[9] E. Marinari AND G. Parisi, *Simulated Tempering: a New Monte Carlo Scheme*, Europhysics Letters, 19 (1992), pp. 451-458.

[10] J. I. Siepmann AND D. Frenkel, *Configuration bias Monte Carlo: a new sampling scheme for flexible chains*, Molecular Physics, 75(1) (1992), pp. 59-70.

[11] C. J. Geyer AND E. A. Thompson, *Annealing Markov Chain Monte Carlo with Applications to Ancestral Inference*, J. of the American Statistical Association, 90 (1995), pp. 909-920.

[12] J. A. Vrugt, H. V. Gupta, L. A. Bastidas, W. Boutem, AND S. Sorooshian, *Effective and Efficient Algorithm for Multiobjective Optimization of Hydrologic Models*, Water Resources Research, 39(8) (2002), pp. 1214-1232.

[13] Y. Tang, P. Reed, AND T. Wagener, *How effective and efficient are multiobjective evolutionary algorithms at hydrologic model calibration?* Hydrol. Earth Syst. Sci., 10 (2006), pp. 289-307.

[14] R. Storn AND K. Price, *Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces*, J. of Global Optimization, **11**(4) (1997), pp. 341–359.

[15] M. Koppen AND R. Vicente-Garcia, *A Fuzzy Scheme for the Ranking of Multivariate Data and its Application*, Proceedings of the IEEE Annual Meeting of the North American Fuzzy Information Processing Society, 2004.

[16] K. Deb, *Multi-objective optimization using evolutionary algorithms*, John Wiley&Sons, 2001.

[17] C. J. F. Ter Braak, *A Markov chain Monte Carlo version of the genetic algorithm differential evolution: easy Bayesian computing for real parameter spaces*, Stat. Comput., 16 (2006), pp. 239-249.

[18] J. Goodman AND A. Sokal, *Multigrid Monte Carlo method: Conceptual foundations*, Physical Review D, 40(6) (1989), pp. 2035-2071.

[19] B. A. Berg, *Markov Chain Monte Carlo Simulations and Their Statistical Analysis*, World Scientific, p.197, 2004.

[20] E. Zitzler, K. Deb, AND L. Thiele, *Comparison of Multiobjective Evolutionary Algorithms: Empirical Results*, Evolutionary Computation, 8(2) (2000), pp. 173-195.

[21] E. Zitzler AND L. Thiele, *Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach*, IEEE Transactions on Evolutionary Computation, 3(4) (1999), pp. 257-271.

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

[23] D. A. Levin, Y. Peres, AND E. L. Wilmer, *Markov Chains and Mixing Times*, American Mathematical Society, Providence, RI, 2008.

[24] J. D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krüger, A. E. Lefohn, AND T. Purcell, *A Survey of General-Purpose Computation on Graphics Hardware*, Computer Graphics Forum, 26(1) (2007), pp. 80-113.

[25] W. Zhu, A. Yaseen, AND Y. Li, *DEMCMC-GPU: An Efficient Multi-Objective Optimization Method with GPU Acceleration on the Fermi Architecture*, New Generation Computing, 29(2) (2011), pp. 163-184.

[26] Y. Li, I. Rata, AND E. Jakobsson, *Sampling Multiple Scoring Functions Can Improve Protein Loop Structure Prediction Accuracy*, Journal of Chemical Information and Modeling, 51(7) (2011), pp. 1656-1666.

[27] J. Michal, J. Dobes, AND D. Cerny, *Multiobjective optimization with an asymptotically uniform coverage of Pareto front*, Proceedings of IEEE International Symposium on Circuits and Systems, 2010.

[28] C. A. Coello Coello, G. B. Lamont, D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer, Berlin, Germany, 2007.

[29] K. Deb, M. Mohan, AND S. Mishra, *Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solution.* Evol. Comput., 13(4) (2005), pp. 501–502.

[30] Q. F. Zhang AND H. Li, *MOEA/D: a multiobjective evolutionary algorithm based on decomposition*, IEEE Trans. Evol. Comput., 11(6) (2007), pp. 712–731.
[31] A. Auger, J. Bader, D. Brockhoff, AND E. Zitzler, *Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications*, Theoretical Computer Science, 425 (2012), pp. 75-103.
[32] D. Hadka AND P. Reed, Borg: An Auto-Adaptive Many-Objective Evolutionary Computing Framework, Evol. Comput., (2012), pp. 1-30.
[33] J. A. Vrugt AND B. A. Robinson, *Improved evolutionary optimization from genetically adaptive multimethod search*, Proc. Natl. Acad. Sci., 104 (2007), pp. 708-711.