Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 949480, 13 pages
http://dx.doi.org/10.1155/2015/949480

Research Article

Spacecraft Multiple-Impulse Trajectory Optimization Using Differential Evolution Algorithm with Combined Mutation Strategies and Boundary-Handling Schemes

Yuehe Zhu, Hua Wang, and Jin Zhang

College of Aerospace Science and Engineering, National University of Defense Technology, Changsha, Hunan 410073, China

Correspondence should be addressed to Jin Zhang; zhangjin@nudt.edu.cn

Received 25 January 2015; Revised 7 April 2015; Accepted 14 April 2015

Academic Editor: Yakov Strelniker

Since most spacecraft multiple-impulse trajectory optimization problems are complex multimodal problems with boundary constraint, finding the global optimal solution based on the traditional differential evolution (DE) algorithms becomes so difficult due to the deception of many local optima and the probable existence of a bias towards suboptimal solution. In order to overcome this issue and enhance the global searching ability, an improved DE algorithm with combined mutation strategies and boundary-handling schemes is proposed. In the first stage, multiple mutation strategies are utilized, and each strategy creates a mutant vector. In the second stage, multiple boundary-handling schemes are used to simultaneously address the same infeasible trial vector. Two typical spacecraft multiple-impulse trajectory optimization problems are studied and optimized using the proposed DE method. The experimental results demonstrate that the proposed DE method efficiently overcomes the problem created by the convergence to a local optimum and obtains the global optimum with a higher reliability and convergence rate compared with some other popular evolutionary methods.

1. Introduction

The subject of spacecraft trajectory optimization has a long history [1]. In its early stage, spacecraft trajectory was primarily optimized using analytical theory and gradient-based optimization algorithms. Hughes et al. [2] concluded that these optimization approaches were efficient in some simple cases but did not work well on more complex ones, such as spacecraft multiple-impulse trajectory optimization problems, because most of these cases present complex multimodal peculiarities, which easily converge to the local optimum and make the search of the global optimum difficult.

Since the 1990s, evolutionary algorithms (EAs) have rapidly emerged. Varieties of EAs have been used for spacecraft multiple-impulse trajectory optimization problems. These include genetic algorithm (GA), simulated annealing (SA), differential evolution (DE), and particle swarm optimization (PSO) [3]. Initial applications of EAs to space trajectory optimization mainly employed GAs in conjunction with gradient-based methods [4–6]. A combination of artificial neural networks with EAs has also been applied to solar sail trajectories [7]. Sentinella and Casalino [8] mentioned that the use of EAs for low-thrust trajectory optimization is less attractive due to the large number of variables required to describe low-thrust trajectories with sufficient accuracy, while EAs are better suited to the optimization of impulse trajectories which can be described by a limited number of variables and the number of function evaluations that are required to obtain the optimal solution is usually acceptable.

In this paper, we focus our research on two typical multiple-impulse trajectory optimization problems which are called same-circle rendezvous problem and deep space gravity assist maneuvers problem, respectively. Some researchers have studied these problems based on EAs recently. Luo et al. [9, 10] applied an improved SA method to optimize the multiple-impulse rendezvous problem. Pontani et al. [11] used a PSO method to solve similar cases. Vinkó and Izzo [12] formulated a model of gravity assist using deep space

maneuvers and applied several kinds of EAs to optimize the multiple-impulse gravity assist problem. Vasile et al. [13] furthered the work of Vinkó and Izzo [12] by applying a hybrid algorithm with the DE method and monotonic basin hopping to the original model. Gad and Abdelkhalik [14, 15] presented a software tool based on their developed hidden genes GA and dynamic-size multiple population GA method to solve the deep space gravity assist maneuvers problems. These employed EAs tend to have a stronger searching ability than traditional approaches but cannot yet converge to the global optimum when optimizing multimodal spacecraft multiple-impulse trajectory problems. Therefore, a more powerful and efficient EA is necessary in order to search the global optimum when solving these problems.

The DE algorithm, as a simple yet efficient optimizer with fewer parameters, is currently one of the most widely used EAs. First proposed by Storn and Price [17, 18], it has received a substantial amount of attention. Many researchers have studied DE and proposed numerous notable variants which have been verified on a series of numerical problems [16, 19–21]. Thanks to its high efficiency, DE has been applied to solve optimization problems in many fields, including the spacecraft trajectory optimization [8, 12, 13, 22–25]. However, because only one search operator is used in most modified DE variants, they may have excellent optimization performance on unimodal problems but their capability to solve the multimodal problems is not so outstanding, especially for the complex multiple-impulse trajectory optimization problem. Hence, some researchers applied multiple search operators in the algorithm. Tasgetiren et al. [26] developed an ensemble DE in such a way that each individual was assigned to one of two distinct mutation strategies or a variable parameter search. Elsayed et al. [27] proposed an improved differential evolution algorithm that uses a mix of different mutation operators to solve the benchmark and some real-world optimization problems.

Moreover, for the problems with boundary constraint, the boundary handling also plays an important role in the evolutionary process except the mutation operation. Different boundary-handling schemes may have discrepant peculiarities and significantly affect the optimization performance. Xu and Rahmat-Samii [28] analysed the effect of different boundary-handling schemes in the PSO method, and multiple boundary-handling schemes were applied by Huang and Mohan [29] in order to improve the algorithm's robustness. However, little literature has reported the usage of different boundary-handling schemes on DE's multiple search operators.

In our research, in order to further enhance the optimization performance of DE algorithm and then to obtain global optimal solutions of the spacecraft multiple-impulse trajectory optimization problems, we studied not only multiple mutation strategies but also multiple boundary-handling schemes and extended the traditional DE method to a modified variant with combined mutation strategies and boundary-handling schemes. By comparing our modified DE method with some other popular evolutionary methods in amount of simulations, we found that the global searching ability of DE algorithm could be efficiently improved by simultaneously combining multiple mutation strategies and boundary-handling schemes when solving spacecraft multiple-impulse trajectory optimization problems. Furthermore, we successfully obtain the global optima for the same-circle rendezvous case and deep space gravity assist maneuvers case using the proposed DE method.

Figure 1: Same-circle rendezvous.

The reminder of the paper is organized as follows. In Section 2, the model Description of the Optimization problems is given out. The modification mechanism of the improved DE method is detailed in Section 3. In Section 4, the simulation results are presented first and the comparison and analysis of the employed algorithms followed then. Finally, conclusions are presented in Section 5.

2. Model Description of the Optimization Problems

2.1. Same-Circle Rendezvous. The same-circle rendezvous problem is a multiple-impulse multiple-revolution rendezvous problem. In this, a chaser spacecraft and a target spacecraft have the same circular orbit. The chaser has an initial separation angle of 180° behind the target, which is shown in Figure 1. The chaser must rendezvous with the target at a prescribed time using impulse maneuvers. This problem, using a transfer time of $2.3 * T$ (T is the period of the circular orbit), was first presented by Prussing and Chiu [30] with a four-impulse optimum solution. Colasurdo and Pastrone [31] and Prussing [32] further obtained a better four-impulse optimum solution using the method based on Lawden's theory and gradient-based optimization algorithms. Recently, Luo et al. [10] and Pontani et al. [11] produced similar results using the global convergence ability of the evolutionary algorithms, in which a parallel simulated annealing that utilizes a simplex method was presented in Luo et al. [10] and a PSO method was applied in Pontani et al. [11].

The variables of this problem are

$$\mathbf{X} = \left[T_1, T_2, T_3, T_4, \Delta V_1, \alpha_1, \beta_1, \Delta V_2, \alpha_2, \beta_2\right]^T, \quad (1)$$

in which T_1, T_2, T_3, and T_4 are the time-applying impulses, ΔV_1 and ΔV_2 are the characteristic velocities of the first two impulses, and α_1, β_1 and α_2, β_2 determine the direction of the first two impulses. The moduli and direction of the last two impulses, ΔV_3 and ΔV_4, are determined using Lambert algorithm.

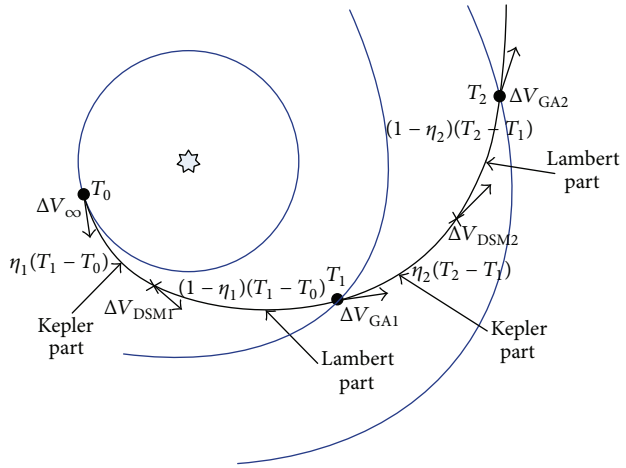

Figure 2: MGA_1DSM flight process.

An optimization will minimize the propellant cost, which is equivalent to minimizing the total characteristic velocity. The objective function is

$$J = \Delta V_1 + \Delta V_2 + \Delta V_3 + \Delta V_4. \tag{2}$$

2.2. Deep Space Gravity Assist Maneuvers. Gravity assist maneuvers can be divided into two categories, which are called multiple gravity assist (MGA) maneuvers and multiple gravity assist using deep space maneuvers (MGA-1DSM). Their corresponding optimization models have been well formulated by Vinkó and Izzo [12]. The MGA manoeuver represents the interplanetary trajectory of a spacecraft equipped with chemical propulsion that can only thrust during its planetocentric phases. This type of problem is easier to optimize because it has fewer design variables. The MGA-1DSM refers to the model in which a spacecraft is able to thrust its engine once at any time between each trajectory leg. Generally, for the same gravity assist sequence, using the MGA-1DSM model is able to design a better trajectory compared with the MGA model. However, there are more variables in the MGA-1DSM model and the problem becomes much more difficult considering the large number of local optima.

In the MGA-1DSM model, the trajectory between two gravity assist maneuvers is divided into two parts, as can be seen in Figure 2. Kepler algorithm is used to propagate the first part, and the second part should be solved by Lambert algorithm. The variables of the MGA-1DSM model are as follows:

$$\begin{aligned} X = \big[&T_0, V_\infty, u, v, \eta_1, T_1, r_{p2}, i_{b2}, \eta_2, T_2, \ldots, r_{pN-1}, \\ &i_{bN-1}, \eta_{N-1}, T_{N-1}\big]^T, \end{aligned} \tag{3}$$

where T_0 is the departure time. Here, V_∞, u, and v define the heliocentric direction of the departure hyperbolic velocity, T_i is the time rendezvous with each planet, η_i is a coefficient between 0 and 1 that determines the flight time of the Kepler part, and r_{pi} and i_{bi} are used to calculate each velocity increment $\Delta\mathbf{V}_{GA}$ as follows:

$$\begin{aligned} \Delta\mathbf{V}_{in} &= \mathbf{V}_{in} - \mathbf{V}_{pla}, \\ e &= 1 + \frac{r_{pi}}{\mu_{pla}\left\|\Delta\mathbf{V}_{in}\right\|^2}, \\ \delta &= 2\arcsin\left(\frac{1}{e}\right), \\ \mathbf{i}_x &= \frac{\Delta\mathbf{V}_{in}}{\left\|\Delta\mathbf{V}_{in}\right\|}, \\ \mathbf{i}_y &= \mathbf{i}_x \times \frac{\mathbf{r}_{pla}}{\left\|\mathbf{i}_x \times \mathbf{r}_{pla}\right\|}, \\ \mathbf{i}_z &= \mathbf{i}_x \times \mathbf{i}_y, \\ \frac{\Delta\mathbf{V}_{out}}{\left\|\Delta\mathbf{V}_{out}\right\|} &= \cos\delta\mathbf{i}_x + \sin i_{bi}\sin\delta\mathbf{i}_y + \cos i_{bi}\sin\delta\mathbf{i}_z, \\ \mathbf{V}_{out} &= \mathbf{V}_{pla} + \Delta\mathbf{V}_{out}, \\ \Delta\mathbf{V}_{GA} &= \mathbf{V}_{out} - \mathbf{V}_{in}, \end{aligned} \tag{4}$$

where $\mathbf{V}_{pla}$ is the velocity of planet in heliocentric ecliptic inertial reference frame (HEIRF), $\mathbf{V}_{in}$ and $\mathbf{V}_{out}$ are the velocities of spacecraft before and after flying by the planet in HEIRF, and μ_{pla} is the planet gravitational constant.

Optimization also minimizes the total characteristic velocity:

$$J = V_\infty + \sum \Delta V_{DSMi} + \Delta V_{final}, \tag{5}$$

where V_∞ is the escape velocity from Earth, ΔV_{DSMi} represent the middle impulses between each two gravity assist times, and ΔV_{final} is the final impulse used to rendezvous with the target planet.

3. Differential Evolution Algorithm with Combined Mutation Strategies and Boundary-Handling Schemes (DE_CMSBHS)

3.1. Classic DE Algorithm. The DE algorithm is a simple real parameter optimization algorithm [33]. It proceeds through a simple cycle of stages, which include mutation, crossover, boundary handling, and selection.

3.1.1. Population Initialization. Given a function $f(\mathbf{X})$ with D-dimensional variables, the population is initialized with a size of. The ith vector of the population $\mathbf{X}_i$ for each generation can be expressed as

$$\mathbf{X}_i = \left[x_{i1}, x_{i2}, x_{i3}, \ldots, x_{iD}\right]^T, \quad i = 1, 2, 3, \ldots, Np. \tag{6}$$

The search space is constrained by the prescribed minimum and maximum bounds:

$$\mathbf{X}_{\min} = [x_{\min 1}, x_{\min 2}, x_{\min 3}, \ldots, x_{\min D}]^T,$$
$$\mathbf{X}_{\max} = [x_{\max 1}, x_{\max 2}, x_{\max 3}, \ldots, x_{\max D}]^T. \quad (7)$$

The jth component of the ith vector in (6) is initialized as follows:

$$x_{ij} = x_{\min j} + \text{rand}_{ij}[0,1] \cdot \left(x_{\max j} - x_{\min j}\right), \quad j = 1, 2, 3, \ldots, D, \quad (8)$$

where $\text{rand}_{ij}[0,1]$ is a random number uniformly distributed between 0 and 1.

3.1.2. Mutation Operation. After initialization, each individual vector $\mathbf{X}_i$, called a target vector, employs the mutation operation in order to create a corresponding mutant vector $\mathbf{V}_i = [v_{i1}, v_{i2}, v_{i3}, \ldots, v_{iD}]^T$ according to the prescribed mutation strategy. The following mutation strategies are frequently employed in the classic DE.

"DE/rand/1:"

$$\mathbf{V}_i = \mathbf{X}_{r1} + F_i \cdot \left(\mathbf{X}_{r2} - \mathbf{X}_{r3}\right). \quad (9)$$

"DE/rand/2:"

$$\mathbf{V}_i = \mathbf{X}_{r1} + F_i \cdot \left(\mathbf{X}_{r2} - \mathbf{X}_{r3}\right) + F_i \cdot \left(\mathbf{X}_{r4} - \mathbf{X}_{r5}\right). \quad (10)$$

"DE/best/1:"

$$\mathbf{V}_i = \mathbf{X}_{\text{best}} + F_i \cdot \left(\mathbf{X}_{r1} - \mathbf{X}_{r2}\right). \quad (11)$$

"DE/best/2:"

$$\mathbf{V}_i = \mathbf{X}_{\text{best}} + F_i \cdot \left(\mathbf{X}_{r1} - \mathbf{X}_{r2}\right) + F_i \cdot \left(\mathbf{X}_{r3} - \mathbf{X}_{r4}\right). \quad (12)$$

"DE/current-to-rand/1:"

$$\mathbf{V}_i = \mathbf{X}_i + F_i \cdot \left(\mathbf{X}_{r1} - \mathbf{X}_i\right) + F_i \cdot \left(\mathbf{X}_{r2} - \mathbf{X}_{r3}\right). \quad (13)$$

"DE/rand-to-best/1:"

$$\mathbf{V}_i = \mathbf{X}_{r1} + F_i \cdot \left(\mathbf{X}_{\text{best}} - \mathbf{X}_{r1}\right) + F_i \cdot \left(\mathbf{X}_{r2} - \mathbf{X}_{r3}\right). \quad (14)$$

"DE/current-to-best/1:"

$$\mathbf{V}_i = \mathbf{X}_i + F_i \cdot \left(\mathbf{X}_{\text{best}} - \mathbf{X}_i\right) + F_i \cdot \left(\mathbf{X}_{r1} - \mathbf{X}_{r2}\right). \quad (15)$$

The indices $r1, r2, r3, r4$, and $r5$ are five different integers, not equal to i randomly selected from $[1, Np]$. We use $\mathbf{X}_{\text{best}}$ to denote the best individual vector of the current generation, and F_i is the mutation factor of the ith vector. The mutation factor is an important parameter in the algorithm. In the classical DE method, F_i is a fixed value within the range $[0, 1]$.

3.1.3. Crossover Operation. Crossover is applied after the mutation operation. Crossover gives the new individual vector $\mathbf{U}_i = [u_{i1}, u_{i2}, u_{i3}, \ldots, u_{iD}]^T$, which is called a trial vector, the genes from both the target vector $\mathbf{X}_i$, and the mutant vector $\mathbf{V}_i$. Binomial crossover is more frequently used in the crossover operation:

$$u_{ij} = \begin{cases} v_{ij}, & \text{if } \text{rand}_{ij}[0,1] \le \text{CR}_i \text{ or } j = j_{\text{rand}}, \\ x_{ij}, & \text{otherwise.} \end{cases} \quad (16)$$

In (16), j_{rand} is an integer randomly selected from $[1, D]$. This ensures that at least one component in $\mathbf{U}_i$ comes from $\mathbf{V}_i$, which maintains the efficiency of each crossover operation. Here, CR_i is the crossover probability of the ith vector. This crossover probability is another important parameter in the algorithm and has a fixed value within the range $[0, 1]$ in the classical DE method.

3.1.4. Boundary Handling. In most problems, especially real-world problems, the vector $\mathbf{X}_i$ is restricted by the boundary constraint. Hence, the trial vector $\mathbf{U}_i$ should be in the range $[\mathbf{X}_{\min}, \mathbf{X}_{\max}]$. If $u_{ij} < x_{\min j}$ or $u_{ij} > x_{\max j}$, boundary handling must be applied in order to address the jth component's presence in the range. The common boundary-handling scheme recreates a new component, as in (4), to replace the original one.

3.1.5. Selection. After the previous steps have been completed, each trial vector must be evaluated. A "greedy" strategy is applied in the selection operation. This means that as long as $(\mathbf{U}_i) \le f(\mathbf{X}_i)$, then the vector $\mathbf{U}_i$ enters the next generation. Otherwise, $\mathbf{X}_i$ is retained in the next generation.

3.2. Combined Mutation Strategies. Mutation is the most important step in a DE algorithm. Equations (9)–(15) are some efficient mutation strategies frequently used in various DE methods. The effectiveness of some other strategies, such as the triangular mutation strategy proposed by Fan and Lampinen [34], have also been verified. In general, all mutation strategies can be divided into two classes. One group emphasizes the global search, similar to the "DE/rand/1," "DE/rand/2," and "DE/current-to-rand/1" strategies. The other class focuses on the local search. Examples of this include the "DE/best/1," "DE/best/2," and "DE/current-to-best/1" strategies.

For complex optimization problems, especially multimodal problems, using one of these strategies alone cannot ensure that the algorithm's global and local searches are effective and efficient. Qin et al. [20] employed multiple mutation strategies in the SADE method. A strategy is chosen based on its probability, which is updated according to the success ratios of the past generations. This approach is self-adaptive to a certain extent. However, only one of the available strategies is used in each mutation operation. These strategies are not used to their full potential, and choosing a strategy based on its probability is an unreliable approach. For a given target vector in a current generation, the strategy with a

higher success probability may not create a better trial vector than one with a lower success probability.

In this case, in order to fully utilize the advantages of multiple mutation strategies, combined mutation strategies are applied in DE_CMSBHS to simultaneously improve the global and local searches. Testing demonstrates that the combination of the following four strategies produces a better performance: "DE/rand/1," "DE/rand/2," "DE/current-to-rand/1," and "DE/current-to-*p*best/1." The "DE/current-to-*p*best/1" strategy is used in the JADE algorithm, which was proposed by Zhang and Sanderson [21]. The *p*best vector is randomly chosen from the top p% of individuals in the population. According to the conclusions in Zhang and Sanderson [21], the parameter p in the range [5, 20] performs better. We used $p = 20$ in the TP_SDE algorithm in order to enhance its global search ability.

The first three global searching strategies can help maintain the population's diversity to some degree; the fourth local searching strategy can improve the algorithm's search speed and accuracy. Even when the fourth strategy converges to a local optimum, the first three strategies increase the possibility of escaping the local optimum in order to evolve to the better solution. Their relationship can be regarded as complementation. However, if the fourth strategy is replaced with "DE/best/1" or "DE/current-to-best/1," the algorithm's optimization performance significantly worsens. Because these two "best" strategies are too greedy, the first three strategies will be overwhelmed by the "best" strategy and become useless. Once the algorithm converges to the local optimum, it cannot progress further. Thus, we apply the less greedy strategy, "DE/current-to-*p*best/1." On one hand, this strategy can decrease the probability of premature convergence. On the other hand, the first three strategies will be used more frequently in the evolution procedure in order to maintain the population's diversity. The comparisons made in Section 4 demonstrate that the optimization performance of combined mutation strategies is better than methods that only use a single strategy.

3.3. Combined Boundary-Handling Schemes. When addressing problems with boundary constraints, boundary handling is an important operation if a trial vector $\mathbf{U}_i$ exceeds the range. Because boundary handling essentially repeats the initialization process in order to replace the infeasible component, the boundary-handling scheme determines the distribution of the new trial vector $\overline{\mathbf{U}_i}$ in the design space. Different schemes can significantly affect the final result.

Currently, only one scheme, which is the same as the one in (8), is used to operate boundary handling in most evolutionary algorithms. Randomly creating a new component with this scheme does not appear to be sufficiently efficient.

Although an individual with a component that exceeds the range is not a feasible solution, it is still valuable because it contains information about the mutation. In order to fully utilize this information about the current and infeasible individuals, multiple boundary-handling schemes are applied in the DE_CMSBHS process. Testing demonstrates that the combination of the following four schemes performs better.

Scheme 1 ("whole_rand"). Consider

$$\begin{aligned} &\text{if } \left(u_{ij} < x_{\min j} \text{ or } u_{ij} > x_{\max j}\right), \\ &\overline{u}_{ij} = x_{\min j} + \text{rand}_{ij}\,[0,1] \cdot \left(x_{\max j} - x_{\min j}\right). \end{aligned} \tag{17}$$

Scheme 2 ("current_rand"). Consider

$$\begin{aligned} &\text{if } \left(u_{ij} < x_{\min j}\right), \\ &\qquad \overline{u}_{ij} = x_{\min j} + \text{rand}_{ij}\,[0,1] \cdot \left(x_{ij} - x_{\min j}\right), \\ &\text{if } \left(u_{ij} > x_{\max j}\right), \\ &\qquad \overline{u}_{ij} = x_{\max j} - \text{rand}_{ij}\,[0,1] \cdot \left(x_{\max j} - x_{ij}\right). \end{aligned} \tag{18}$$

Scheme 3 ("reflect_rand"). Consider

$$\begin{aligned} &\text{if } \left(u_{ij} < x_{\min j}\right), \\ &\qquad \overline{u}_{ij} = x_{\min j} + \text{rand}_{ij}\,[0,1] \cdot \left(u'_{ij} - x_{\min j}\right), \\ &\qquad u'_{ij} - x_{\min j} = x_{\min j} - u_{ij}, \\ &\text{if } \left(u_{ij} > x_{\max j}\right), \\ &\qquad \overline{u}_{ij} = x_{\max j} - \text{rand}_{ij}\,[0,1] \cdot \left(x_{\max j} - u'_{ij}\right), \\ &\qquad u_{ij} - x_{\max j} = x_{\max j} - u'_{ij}. \end{aligned} \tag{19}$$

Scheme 4 ("cut_off"). Consider

$$\begin{aligned} &\text{if } \left(u_{ij} < x_{\min j}\right), \quad \overline{u}_{ij} = x_{\min j}, \\ &\text{if } \left(u_{ij} > x_{\max j}\right), \quad \overline{u}_{ij} = x_{\max j}. \end{aligned} \tag{20}$$

In (17)–(20), u_{ij} is the before-handling component and $\overline{u}_{ij}$ is the after-handling component. Figure 3 illustrates these four kinds of boundary-handling schemes. And because the handling schemes of a component exceeding the minimum or maximum are similar, the situation of $u_{ij} > x_{\max j}$ is considered as an example in the following illustration.

As is shown in Figure 3, "whole_rand" is the retaining scheme used to ensure the potential selection of any value within the given range. The "current_rand" scheme randomly creates a new component between the current component and the maximum, which efficiently utilizes the information of the current individual. The "reflect_rand" scheme first creates an u'_{ij}. u'_{ij}, in which the u_{ij} are symmetric with respect to $x_{\max j}$. Then a new component is randomly created between u'_{ij} and $x_{\max j}$. This scheme efficiently uses the information of infeasible individuals. The "cut_off" scheme directly replaces the component with the maximum, which is extremely efficient if some components of the optimum solution are on the boundary. The comparisons made in Section 4 also demonstrate a better optimization performance of combined boundary-handling schemes than that of methods that use a single scheme.

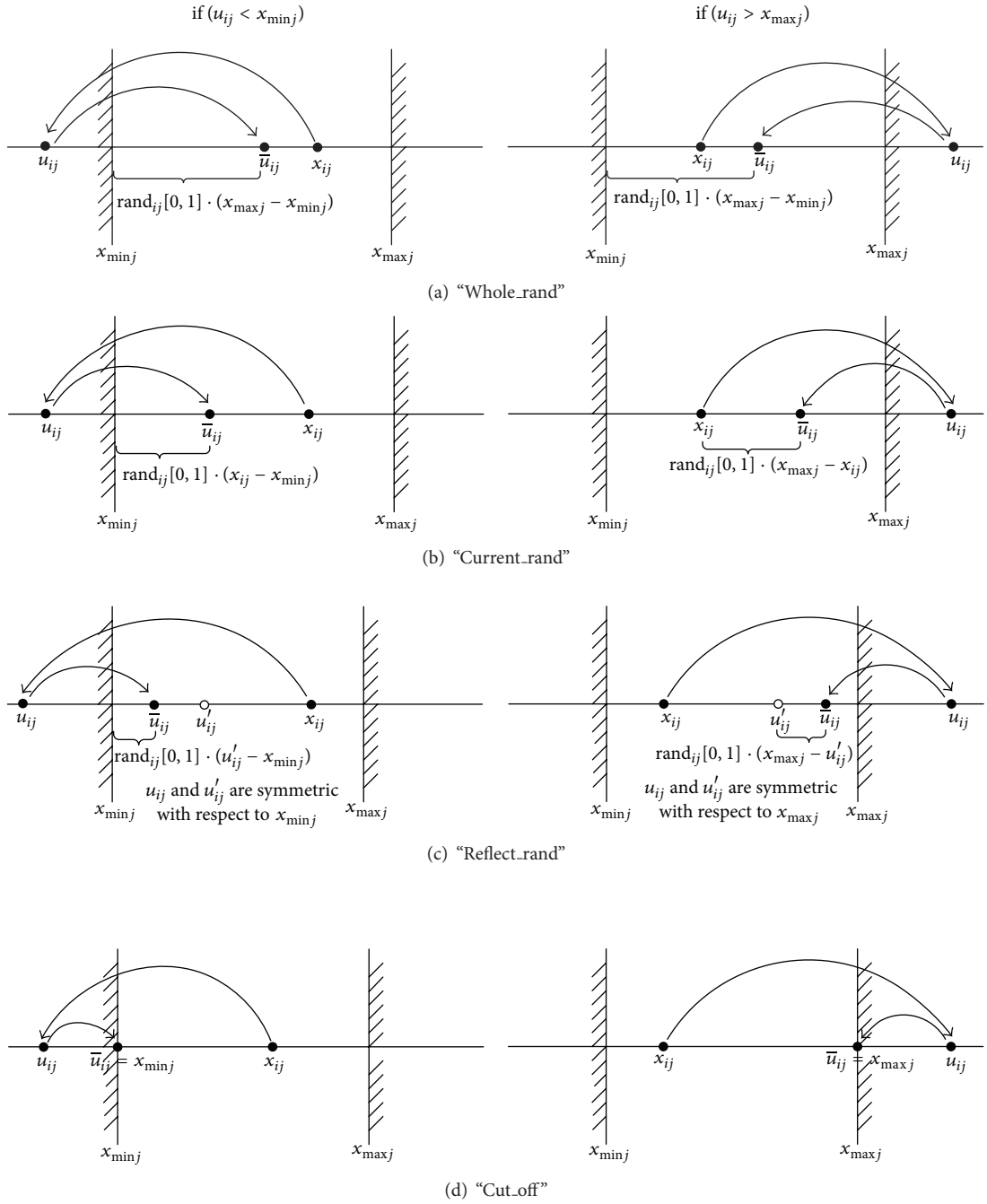

Figure 3: Illustration of the four kinds of boundary-handling schemes.

3.4. Self-Adaption of Parameters. The parameters F and CR in the classic DE algorithm are usually determined by testing or experience. They are always fixed values in any evolution procedure. However, the efficiency and reliability of this situation seems insufficient. For this reason, many studies on the self-adaption of parameters exist. The self-adaption of parameters scheme applied in this paper refers to the one proposed by Islam et al. [16]. The parameter F adapts as follows:

$$F_i = \text{Cauchy}\left(\mu F_G, 0.1\right). \tag{21}$$

In (21), F_i is a random number selected from a Cauchy distribution with mean and standard deviation 0.1. Here, μF_G is updated in every generation in accordance with

$$\mu F_{G+1} = \omega_F \cdot \mu F_G + \left(1 - \omega_F\right) \cdot \text{mean}\left(S_F\right), \tag{22}$$

where ω_F is a random number uniformly distributed between 0.8 and 1, S_F is the assemblage of all successful mutation factors at the current generation, and mean(S_F) represents the

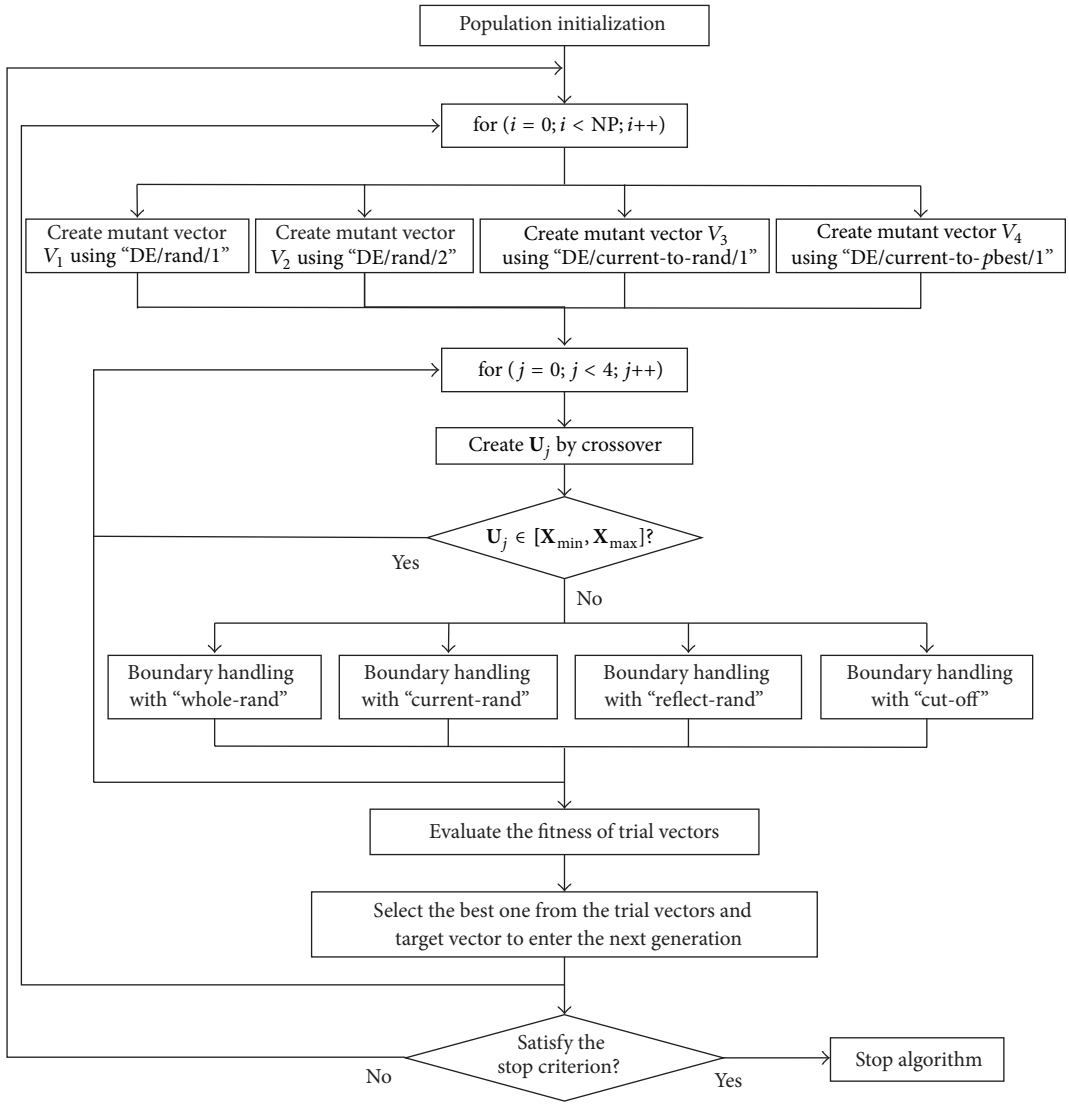

Figure 4: DE_CMSBHS algorithm flow.

power mean of all successful mutation factors, which can be calculated by

$$\text{mean}\left(S_F\right) = \sum_{x \in S_F} \left(\frac{x^m}{|S_F|} \right)^{1/m}, \quad m = 2. \tag{23}$$

In (23), $|S_F|$ is the number of successful mutation factors.

The self-adaption scheme used for CR is similar to the one used for F. The difference is that CR_i is a random number taken from the Gaussian distribution

$$\begin{aligned} \text{CR}_i &= \text{Gaussian}\left(\mu\text{CR}_G, 0.1\right), \\ \mu\text{CR}_{G+1} &= \omega_{\text{CR}} \cdot \mu\text{CR}_G + \left(1 - \omega_{\text{CR}}\right) \cdot \text{mean}\left(S_{\text{CR}}\right), \\ \text{mean}\left(S_{\text{CR}}\right) &= \sum_{x \in S_{\text{CR}}} \left(\frac{x^n}{|S_{\text{CR}}|} \right)^{1/n}, \quad n = 1.5. \end{aligned} \tag{24}$$

The fundamental idea of this self-adaption scheme is that the parameters of the next generation are guided according to the parameters of all successful scale factors in the current generation. The parameter F selected from the Cauchy distribution maintains the population's diversity, and CR, chosen from the Gaussian distribution, concentrates the value. If the value for F or CR exceeds the range $(0, 1]$, it will be recreated until the value is within the range. If μF_G and μCR_G are 0.5 and 0.75, respectively, at the first generation, then they are adapted according to the situation of the following generations. The values for ω_F and ω_{CR}, selected from the uniform distribution $[0.8, 1]$, maintain updating robustness. The parameters m and n do not have fixed values. Numerous experiments show that $m = 2$ and $n = 1.5$ are the best choices for complex multimodal problems.

3.5. Algorithm Flow of DE_CMSHBS. Figure 4 presents the DE_CMSHBS algorithm flow. The stop criterion can be set according to the problem's need. In general, the algorithm stops when the prescribed fitness evaluation number approaches or converges to a plateau in which successive

Table 1: Design boundaries of same-circle rendezvous model.

Variable	LB	UB	Units
T_1	0	0.1	$2.3 * T$
T_2	0.1	0.5	$2.3 * T$
T_3	0.5	0.9	$2.3 * T$
T_4	0.9	1	$2.3 * T$
ΔV_1	0	1500	km/s
α_1	−PI	PI	rad
β_1	−PI	PI	rad
ΔV_2	0	1500	km/s
α_2	−PI	PI	rad
β_2	−PI	PI	rad

iterations no longer produce better results. If attaining the prescribed fitness evaluation number is the stop criterion, fewer evolutionary generations are needed because more fitness evaluation numbers in each generation are contained in the DE_CMSHBS algorithm than those contained in other algorithms.

4. Simulation and Comparison

4.1. Simulation of Same-Circle Rendezvous Case

4.1.1. Result Presentation. Based on the same-circle rendezvous model that is formulated in Section 2, the circular orbit is set to 400 km in height, which is the same as the simulation condition in Luo et al. [10]. The boundaries of each variable are established in Table 1.

The best solution obtained by the DE_CMSBHS method is $\mathbf{X} = [0.000000, 0.207320, 0.792680, 1.000000, 392.546673, 0.500000, 0.547843, 235.587940, 0.500000, 0.142865]^T$ with an objective function value of 1256.27 m/s, surpassing the best solution of Luo et al. [10] using PSASM method with an objective function value of $J = 1256.32$ m/s. Each impulse time and characteristic velocity optimized by DE_CMSBHS and PSASM method are listed in Table 2. The chaser's rendezvous trajectory optimized by the DE_CMSBHS method is given out in Figure 5.

Figure 6 illustrated the primer-magnitude time history of the obtained best solution. Apparently it satisfies Lawden's conditions, which is a necessary condition for an optimal transfer orbit [35]. It indicates that DE_CMSBHS has successfully obtained the optimum solution of the same-circle rendezvous problem.

4.1.2. Algorithm Comparison and Analysis. Except for the DE_CMSBHS method, we also tried eight single-combined DEs in solving the same-circle rendezvous problem to compare with the PSASM method that is used by Luo et al. [10], including four kinds of DEs with combined mutation strategies and four kinds of DEs with combined boundary-handling schemes. For convenience, the single-combined DEs are abbreviated as follows. If we use DE_CMS + "whole_rand" and "DE/rand/1" + DE_CBHS, for instance, DE_CMS + "whole_rand" represents the combination of the

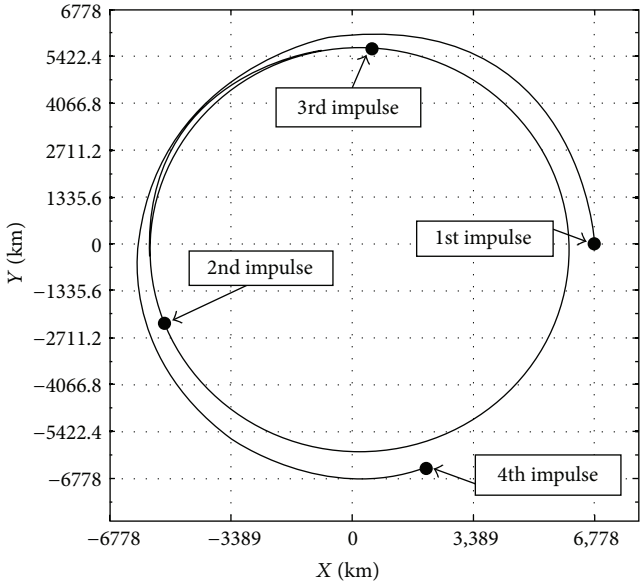

Figure 5: The trajectory of same-circle rendezvous optimized by DE_CMSBHS.

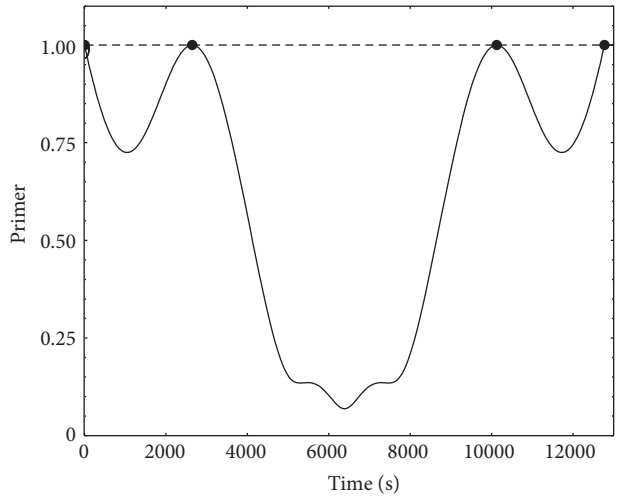

Figure 6: Primer-magnitude time history.

combined mutation strategies with the boundary-handling scheme "whole_rand." "DE/rand/1" + PBHS denotes the combination of the mutation strategy "DE/rand/1" and the combined boundary-handling schemes. The population size NP is uniformly selected to be 100. We carry out 30 independent trials for each method and the results are listed in Table 3. The best solutions of each fitness evaluation numbers (FEs) are in boldface.

Comparing the results in Table 3, several conclusions can be drawn.

(1) In general, the results of all the combined DE methods are better than the PSASM method [10]. Furthermore, the DE_CMSBHS surpasses all other combined DEs.

(2) When the FEs are set to be 3×10^4 and 5×10^4, the mean and standard deviation of the DE_CMSBHS

Table 2: Impulse time and characteristic velocity of the best solution.

Algorithm	Impulses (t_i (s), Δv_i (m/s))				Δv (m/s)
	$i = 1$	$i = 2$	$i = 3$	$i = 4$	
PSASM	(0, 391.48)	(2648.94, 236.51)	(10130.13, 235.83)	(12772.96, 392.50)	1256.32
DE_CMSBHS	(0, 392.55)	(2648.09, 235.59)	(10124.87, 235.59)	(12772.96, 392.55)	1256.27

Table 3: Results of same-circle rendezvous model.

Algorithms	Mean (Std)		
	FEs = 3×10^4	FEs = 5×10^4	FEs = 10^5
PSASM	1428.23 (147.17)	1392.07 (134.85)	1375.98 (114.27)
DE_CMS + "whole_rand"	1264.43 (11.59)	1260.73 (7.58)	**1256.27 (0)**
DE_CMS + "current_rand"	1266.56 (50.38)	1259.49 (6.96)	1262.74 (34.83)
DE_CMS + "reflect_rand"	1259.75 (6.78)	1258.53 (5.28)	**1256.27 (0)**
DE_CMS + "cut_off"	1269.94 (48.65)	1262.35 (33.58)	1262.74 (34.83)
"DE/rand/1" + DE_CBHS	1283.48 (68.35)	1276.48 (66.27)	1275.67 (58.21)
"DE/rand/2" + DE_CBHS	1278.72 (62.06)	1275.72 (58.48)	1269.20 (48.40)
"DE/current-to-rand/1" + DE_CBHS	1269.58 (47.57)	1268.47 (49.41)	1269.20 (48.40)
"DE/current-to-*p*best/1" + DE_CBHS	1261.45 (8.69)	1258.13 (4.15)	**1256.27 (0)**
DE_CMSBHS	**1257.99 (2.89)**	**1256.63 (1.16)**	**1256.27 (0)**

are less than the corresponding values of the combined DEs and PSASM. This indicates that the DE_CMSBHS has a higher optimization speed and is more stable for this problem.

(3) When FEs = 10^5, all of the results converge. In this case, DE_CMSBHS, DE_CMS + "whole_rand," DE_CMS + "reflect_rand," and "DE/current-to-*p*best/1" + DE_CBHS converge to the theoretical optimum point 1256.27 km/s.

It is worth mentioning that there is a local optimum that also satisfies Lawden's conditions. Specifically, the solution is $X = [0.000000, 0.345262, 0.654738, 1.000000, 533.945461, 0.500000, 0.015312, 191.202864, 0.500000, 0.682373]^T$, and the objective function value is 1450.30 m/s. Our experiment shows that results with larger standard deviations in Table 3 converge to this local optimum several times during the 30 trials. However, the DE_CMSBHS, which combines both multiple mutation strategies and boundary-handling schemes, overcomes this challenge and seldom converges to this local solution.

4.2. Deep Space Gravity Assist Maneuvers Case

4.2.1. Result Presentation. Based on the MGA-1DSM model that is formulated in Section 2, we study and optimize the classic Cassini mission. The flight sequence of Cassini mission is Earth–Venus–Venus–Earth–Jupiter–Saturn with 22 variables. The boundaries are established in Table 4.

Table 4: Design boundaries of Cassini mission.

Variable	LB	UB	Units
T_0	−1000	0	MJD2000
V_∞	3	5	km/s
u	0	1	n/a
v	0	1	n/a
T_1	100	400	day
T_2	100	500	day
T_3	30	300	day
T_4	400	1600	day
T_5	800	2200	day
η_1	0.01	0.9	n/a
η_2	0.01	0.9	n/a
η_3	0.01	0.9	n/a
η_4	0.01	0.9	n/a
η_5	0.01	0.9	n/a
r_{p2}	1.05	6	R_{Venus}
r_{p3}	1.05	6	R_{Venus}
r_{p4}	1.15	6.5	R_{Earth}
r_{p5}	1.7	291	R_{Jupiter}
i_{b2}	−pi	pi	rad
i_{b3}	−pi	pi	rad
i_{b4}	−pi	pi	rad
i_{b5}	−pi	pi	rad

This case has been solved by Vasile et al. [13] and Gad and Abdelkhalik [14]. The objective function value of their optimum solution was 8.3889 km/s and 8.3850 km/s, respectively. Solutions optimized by some other researchers are listed on the ESA's official website: http://www.esa.int/gsp/ACT/inf/op/globopt. Currently, the optimum solution on the website is $X = [-779.046754, 3.259114, 0.525976, 0.380865, 167.378952, 424.028254, 53.289741, 589.766955, 2200.000000, 0.769483, 0.513289, 0.027418, 0.263985, 0.599985, 1.348780, 1.050000, 1.307303, 69.809014, -1.593737, -1.959525, -1.554988, -1.513462]^T$ with an objective function value of $J = 8.3832$ km/s.

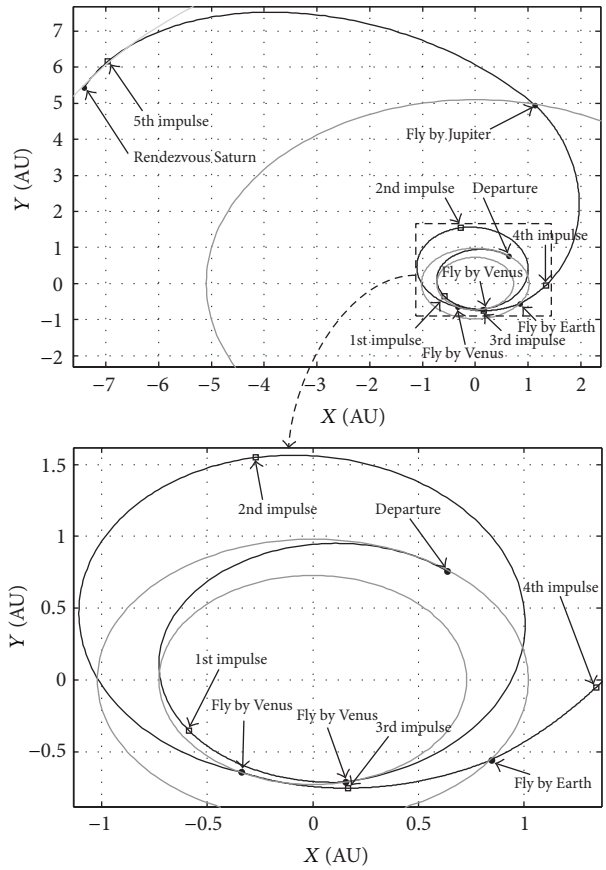

Figure 7: The trajectory of Cassini mission optimized by DE_CMSBHS.

Table 5: The best result of each velocity increment from impulse/gravity assist.

Time (MJD2000)	Impulse/gravity assist	Δv (km/s)
−780.150849	Departure impulse	3.275103
−649.698113	1st impulse	0.463213
−611.660883	Gravity assist by Venus	6.275954
−385.531004	2nd impulse	0.397884
−187.662661	Gravity assist by Venus	6.951561
−166.669404	3rd impulse	0.000000
−134.355766	Gravity assist by Earth	5.108686
−99.8160349	4th impulse	0.000000
455.4159571	Gravity assist by Jupiter	4.442469
2410.306944	5th impulse	0.000001
2655.415957	Rendezvous impulse	4.246699

The best solution obtained by DE_CMSBHS method is $X = [-780.148378, 3.274998, 0.530680, 0.382069, 168.487299, 423.998725, 53.306666, 589.771686, 2200.000000, 0.774442, 0.533183, 0.391920, 0.057816, 0.888942, 1.360604, 1.050000, 1.306803, 69.812308, -1.594186, -1.959564, -1.554776, -1.513431]^T$, which slightly surpasses the current optimum solution with an objective function value of $J = 8.3829$ km/s.

Table 6: Results of Cassini mission solved by some DE variants [16] and optimized by DE_CMSBHS.

Algorithms	Mean (Std)	
	FEs = 5×10^4	FEs = 10^5
DE/rand/1	21.9978 (8.2737)	20.4402 (6.6093)
DE/current-to-best/1	21.5929 (7.4938)	19.8935 (6.7393)
SADE	19.8826 (4.9374)	19.0914 (4.3840)
JADE	19.4690 (4.8837)	18.7155 (4.0283)
MDE_pBX	14.1815 (**3.9485**)	11.2812 (**3.0182**)
DE_CMSBHS	**12.9645** (4.2571)	**10.6425** (3.6447)

Table 7: Results of Cassini mission optimized by the DE_CMS, DE_CBHS, and DE_CMSBHS.

Algorithms	Mean (Std)
	FEs = 2×10^5
DE_CMS + "whole_rand"	12.0145 (5.4773)
DE_CMS + "current_rand"	11.3647 (2.4780)
DE_CMS + "reflect_rand"	13.4766 (4.4237)
DE_CMS + "cut_off"	12.7364 (3.8346)
"DE/rand/1" + DE_CBHS	18.2376 (6.3474)
"DE/rand/2" + DE_CBHS	17.8534 (6.4369)
"DE/current-to-rand/1" + DE_CBHS	17.2675 (5.9437)
"DE/current-to-*p*best/1" + DE_CBHS	15.4394 (5.1473)
DE_CMSBHS	**8.7099 (1.4116)**

Figure 7 presents the optimized trajectory from departure to rendezvousing with Saturn. Each velocity increment from impulse or gravity assist is shown in Table 5. From Table 5 we can find that most characteristic velocity is donated by the initial departure and the final rendezvous. With the help of gravity assist, few velocity increments of the impulses are needed in the middle transfer.

4.2.2. Algorithm Comparison and Analysis. Islam et al. [16] optimized the problem using several DE variants and their results are listed in Table 6. As a comparison for our results that optimized by DE_CMSBHS method, the population size Np was uniformly selected to be 100. Fifty independent trials were performed. Moreover, a comparison between the DE_CMSBHS, four kinds of DE_CMS, and four kinds of DE_CBHS methods are displayed in Table 7.

From Tables 6 and 7, we can make the following observations.

(1) The comparison shows that the results of the DE_CMSBHS method not only exceed the modified DE variants but also the DE_CMS and DE_CBHS method. It verifies that the DE_CMSBHS method has a stronger global search ability for this problem.

(2) The standard deviation (Std) of the DE_CMSBHS method in Table 6 is less than those of the SADE, JADE, and two classic DE methods while it is greater than the MDE_pBX method. This suggests that the optimization stability of the DE_CMSBHS method should be further improved.

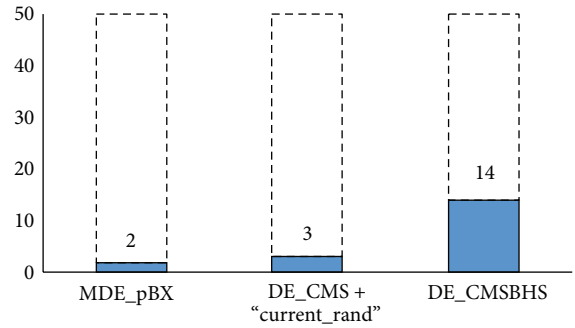

Figure 8: Comparison of convergence numbers between the DE_CMSBHS and other two methods.

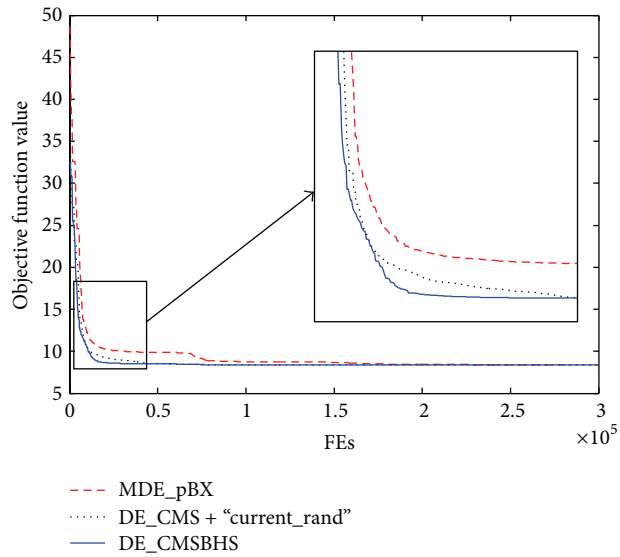

Figure 9: Comparisons of best convergence history between the DE_CMSBHS and other two methods.

(3) In general, the results of all the DE_CMS exceed those of the DE_CBHS method. This can be seen in Table 7. It could be argued that the mutation strategies combination more efficiently improves the DE algorithm than the boundary-handling schemes combination method does.

When optimizing real-world problems, we prefer to obtain an optimum solution. Hence, a further comparison is carried out and the algorithms do not stop until it converges to a plateau, in which successive iterations no longer produce better results. The best results for the compared algorithms in fifty independent trials are listed in Table 8.

It is apparent that the best solutions optimized by MDE_pBX, DE_CMS + "current_rand," and DE_CMSBHS methods surpass the published optimum solution and the DE_CMSBHS's solution is slightly better than other two methods. Figure 8 displays the convergence numbers used to determine the optimum solution of the MDE_pBX, DE_CMS + "current_rand," and DE_CMSBHS methods in the 50 trials. The DE_CMSBHS method converges to the optimum solution in 14 of the trials whereas the MDE_pBX and DE_CMS + "current_rand" methods converge only two and three times, respectively. This indicates that the DE_CMSBHS method has a higher global optimization success rate. The convergence history of the three best results is shown in Figure 9. The graph lets us see that the DE_CMSBHS method converges faster than other two methods.

Table 8: Best results of Cassini mission optimized by the compared algorithms.

Algorithms	Objective function value
DE/rand/1	15.6583
DE/current-to-best/1	11.6284
MDE_pBX	8.3830
DE_CMS + "whole_rand"	9.8373
DE_CMS + "current_rand"	8.3830
DE_CMS + "reflect_rand"	9.0926
DE_CMS + "cut_off"	8.6173
"DE/rand/1" + DE_CBHS	12.8243
"DE/rand/2" + DE_CBHS	12.1534
"DE/current-to-rand/1" + DE_CBHS	11.3536
"DE/current-to-$p$best/1" + DE_CBHS	8.7052
DE_CMSBHS	**8.3829**

In addition, there is a local optimum causing strong interference. The components are $X = [-805.733015, 3.000000, 0.616195, 0.384658, 195.117030, 422.971282, 53.293530, 589.769333, 2200.000000, 0.113728, 0.514959, 0.047143, 0.013736, 0.026443, 1.270316, 1.050000, 1.307191, 69.809127, -1.616293, -1.959523, -1.554919, -1.513431]^T$, and the objective function value is 8.61 km/s. The DE_CMS + "cut-off" method in Table 8 repeatedly converges to this local optimum. Further analysis of this local optimum demonstrates that the second component, V_∞, was selected to be the minimum, which was 3 km/s. Because this component is directly added to the objective function, it seems to be more sensitive to the fitness value than the other components do. Also, because the DE_CMS + "cut –off" method always replaces infeasible components with the lower bound (LB) or upper bound (UB), it clearly increases the possibility of obtaining this local optimum. The DE_CMSBHS method, which simultaneously uses multiple boundary-handling schemes, seldom converges to this local optimum. This indicates that the global search ability of the DE_CMSBHS method is efficiently improved by combining multiple boundary-handling schemes.

5. Conclusion

In order to improve the optimization performance of DE algorithms when solving multimodal spacecraft multiple-impulse trajectory optimization problems, a modified DE variant called DE_CMSBHS which combines four mutation strategies and four boundary-handling schemes is proposed. By applying the DE_CMSBHS method in the same-circle rendezvous problem and deep space gravity assist maneuvers problem, we successfully obtain the global optimum for the

400 km height same-circle rendezvous case and find an optimal solution for the Cassini mission which is currently the best solution as far as I know. Furthermore, we compared the DE_CMSBHS method with eight kinds of single-combined DE methods and some other popular EAs. The simulation results indicate that by simultaneously utilizing multiple mutation strategies and boundary-handling schemes DE_CMSBHS efficiently avoid the issue of converging to the local optimum and tend to the global optimum with a higher optimization speed and success rate.

However, the four mutation strategies and four boundary-handling schemes applied in the DE_CMSBHS method may not produce the most efficient combination for other problems. Future research will focus on a method for determining the most efficient combination according to different problems and will extend the DE_CMSBHS to multiobjective spacecraft trajectory optimization problems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Foundation of China (no. 11402295), the Hunan Provincial Natural Science Foundation of China (no. 2015JJ3020), and the Trans-Century Training Programme Foundation for the Talents by the State Education Commission (NCET-13-0159).

References

[1] B. A. Conway, *Spacecraft Trajectory Optimization*, Cambridge University Press, New York, NY, USA, 2010.

[2] S. P. Hughes, L. M. Mailhe, and J. J. Guzman, "A comparison of trajectory optimization methods for the impulsive minimum fuel rendezvous problem," *Advances in the Astronautical Sciences*, vol. 113, pp. 85–104, 2003.

[3] Y. Z. Luo, J. Zhang, and G. J. Tang, "Survey of orbital dynamics and control of space rendezvous," *Chinese Journal of Aeronautics*, vol. 27, no. 1, pp. 1–11, 2014.

[4] P. Gage, R. Braun, and I. Kroo, "Interplanetary trajectory optimization using a genetic algorithm," *Journal of the Astronautical Sciences*, vol. 43, no. 1, pp. 59–76, 1995.

[5] V. Coverstone-Carroll, "Near-optimal low-thrust trajectories via micro-genetic algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 1, pp. 196–198, 1997.

[6] J. W. Hartmann, V. L. Coverstone-Carroll, and S. N. Williams, "Optimal interplanetary spacecraft trajectories via a Pareto genetic algorithm," *Journal of the Astronautical Sciences*, vol. 46, no. 3, pp. 267–282, 1998.

[7] B. Dachwald and B. Wie, "Solar sail kinetic energy impactor trajectory optimization for an asteroid-deflection mission," *Journal of Spacecraft and Rockets*, vol. 44, no. 4, pp. 755–764, 2007.

[8] M. R. Sentinella and L. Casalino, "Cooperative evolutionary algorithm for space trajectory optimization," *Celestial Mechanics & Dynamical Astronomy*, vol. 105, no. 1–3, pp. 211–227, 2009.

[9] Y.-Z. Luo, G.-J. Tang, Y.-J. Lei, and H.-Y. Li, "Optimization of multiple-impulse, multiple-revolution, rendezvous-phasing maneuvers," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 4, pp. 946–952, 2007.

[10] Y.-Z. Luo, J. Zhang, H.-Y. Li, and G.-J. Tang, "Interactive optimization approach for optimal impulsive rendezvous using primer vector and evolutionary algorithms," *Acta Astronautica*, vol. 67, no. 3-4, pp. 396–405, 2010.

[11] M. Pontani, P. Ghosh, and B. A. Conway, "Particle swarm optimization of multiple-burn rendezvous trajectories," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 4, pp. 1192–1207, 2012.

[12] T. Vinkó and D. Izzo, "Global optimization heuristics and test problems for preliminary spacecraft trajectory design," ACT Technical Report, 2008.

[13] M. Vasile, E. Minisci, and M. Locatelli, "An inflationary differential evolution algorithm for space trajectory optimization," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 2, pp. 267–281, 2011.

[14] A. Gad and O. Abdelkhalik, "Hidden genes genetic algorithm for multi-gravity-assist trajectories optimization," *Journal of Spacecraft and Rockets*, vol. 48, no. 4, pp. 629–641, 2011.

[15] O. Abdelkhalik and A. Gad, "Dynamic-size multiple populations genetic algorithm for multi-gravity-assist trajectories optimization," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 520–529, 2012.

[16] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 482–500, 2012.

[17] R. Storn and K. V. Price, "Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces," Tech. Rep. TR-95-012, ICSI, Berkeley, Calif, USA, 1995.

[18] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[19] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[20] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[21] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transaction on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[22] F. Simeoni and L. Casalino, "Evolutionary optimization of interplanetary trajectories: improvements from initial diversification," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 225, no. 11, pp. 1277–1288, 2011.

[23] D. Izzo, V. M. Becerra, D. R. Myatt, S. J. Nasuto, and J. M. Bishop, "Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories," *Journal of Global Optimization*, vol. 38, no. 2, pp. 283–296, 2007.

[24] M. Vasile, E. Minisci, and M. Locatelli, "A dynamical system perspective on evolutionary heuristics applied to space trajectory optimization problems," in *Proceedings of the IEEE*

Congress on Evolutionary Computation (CEC '09), pp. 2340–2347, May 2009.

[25] B. Addis, A. Cassioli, M. Locatelli, and F. Schoen, "A global optimization method for the design of space trajectories," *Computational Optimization and Applications*, vol. 48, no. 3, pp. 635–652, 2011.

[26] M. F. Tasgetiren, P. N. Suganthan, P. Quan-Ke, R. Mallipeddi, and S. Sarman, "An ensemble of differential evolution algorithms for constrained function optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '10)*, pp. 1–8, Barcelona, Spain, July 2010.

[27] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "An improved self-adaptive differential evolution algorithm for optimization problems," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 89–99, 2013.

[28] S. Xu and Y. Rahmat-Samii, "Boundary conditions in particle swarm optimization revisited," *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 3 I, pp. 760–765, 2007.

[29] T. Huang and A. S. Mohan, "A hybrid boundary condition for robust particle swarm optimization," *IEEE Antennas and Wireless Propagation Letters*, vol. 4, no. 1, pp. 112–117, 2005.

[30] J. E. Prussing and J.-H. Chiu, "Optimal multiple-impulse time-fixed rendezvous between circular orbits," *Journal of Guidance, Control, and Dynamics*, vol. 9, no. 1, pp. 17–22, 1986.

[31] G. Colasurdo and D. Pastrone, "Indirect optimization method for impulsive transfers," in *AIAA/AAS Astrodynamics Conference*, Scottsdale, Ariz, USA, 1994.

[32] J. E. Prussing, "A class of optimal two-impulse rendezvous using multiple-revolution Lambert solutions," *Journal of the Astronautical Sciences*, vol. 48, no. 2-3, pp. 131–148, 2000.

[33] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transaction on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[34] H. Y. Fan and J. Lampinen, "A trigonometric mutation operation to differential evolution," *Journal of Global Optimization*, vol. 27, no. 1, pp. 105–129, 2003.

[35] D. F. Lawden, *Optimal Trajectories for Space Navigation*, Butterworths, London, UK, 1963.