Optimization Methods and Software
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/goms20

Some modifications of low-dimensional simplex evolution and their convergence

Changtong Luo [a] , Shao-Liang Zhang [b] & Bo Yu [c]

[a] Institute of Mechanics, Chinese Academy of Sciences, Beijing, 100190, P.R. China

[b] Department of Computational Science and Engineering, Nagoya University, Nagoya, 464-8603, Japan

[c] Department of Applied Mathematics, Dalian University of Technology, Dalian, 116024, P.R. China

Version of record first published: 01 Sep 2011.

To cite this article: Changtong Luo, Shao-Liang Zhang & Bo Yu (2013): Some modifications of low-dimensional simplex evolution and their convergence, Optimization Methods and Software, 28:1, 54-81

To link to this article: http://dx.doi.org/10.1080/10556788.2011.584876

Optimization Methods & Software
Vol. 28, No. 1, February 2013, 54–81

Taylor & Francis
Taylor & Francis Group

Some modifications of low-dimensional simplex evolution and their convergence

Changtong Luo[a]*, Shao-Liang Zhang[b] and Bo Yu[c]

[a]*Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, P.R. China;* [b]*Department of Computational Science and Engineering, Nagoya University, Nagoya 464-8603, Japan;* [c]*Department of Applied Mathematics, Dalian University of Technology, Dalian 116024, P.R. China*

(*Received 3 February 2010; final version received 26 April 2011*)

Low-dimensional simplex evolution (LDSE) is a real-coded evolutionary algorithm for global optimization. In this paper, we introduce three techniques to improve its performance: low-dimensional reproduction (LDR), normal struggle (NS) and variable dimension (VD). LDR tries to preserve the elite by keeping some of its (randomly chosen) components. LDR can also help the offspring individuals to escape from the hyperplane determined by their parents. NS tries to enhance its local search capability by allowing unlucky individual search around the best vertex of m-simplex. VD tries to draw lessons from recent failure by making further exploitation on its most promising sub-facet. Numerical results show that these techniques can improve the efficiency and reliability of LDSE considerably. The convergence properties are then analysed by finite Markov chains. It shows that the original LDSE might fail to converge, but modified LDSE with the above three techniques will converge for any initial population. To evaluate the convergence speed of modified LDSE, an estimation of its first passage time (of reaching the global minimum) is provided.

Keywords: global optimization; evolutionary algorithm; genetic algorithm; low dimensional; variable dimension; Markov chain

1. Introduction

Global optimization (GO) methods are widely used in different application fields, including economics, financial industry, engineering, modern management, information and control. We consider the GO as follows:

$$\min_{\boldsymbol{x}\in\boldsymbol{\Omega}} f(\boldsymbol{x}),$$

where $\boldsymbol{\Omega} = \{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}\}$, and the objective function f might be non-linear, non-convex, multi-modal, non-smooth, discontinuous, or black-box type.

Since 1960s, different kinds of evolutionary algorithms (EAs) have been proposed [4], including genetic algorithm [9], evolution strategy [17], evolutionary programming [5], differential evolution (DE) [21], particle swarm optimization [11], etc. EAs play an important role in GO,

*Corresponding author. Email: luo@imech.ac.cn

ISSN 1055-6788 print/ISSN 1029-4937 online

http://dx.doi.org/10.1080/10556788.2011.584876
http://www.tandfonline.com

especially in case there are many local optima and/or the gradient information is unavailable, hard to compute, or unreliable (numerically unstable, e.g. in the presence of noise). However, their efficiency (i.e. convergence speed) and reliability (i.e. percentage of success at a given computational cost) are not satisfactory yet, and have become the bottleneck of their applications.

Hybridizing conventional local optimization methods is a good way to speed up EAs. Existing ways of hybridization could be classified into three types: (1) (E + T)-type, in which EA is focused on global exploration, and the conventional optimization algorithm is used for local exploitation. First, EA is applied to find a most promising basin of attraction and get an initial point. Then the conventional algorithm is used to discover the bottom of the basin from the initial point provided by EA. This type of hybridization is very easy to use, so it is widely used in practical engineering computation. For example, the latest Matlab uses a hybrid function to specify a local optimizer (fminsearch, patternsearch, fminunc, or fmincon) that runs after the genetic algorithm terminates [14]. (2) (E + T + E)-type, in which the conventional algorithm is embedded in EA to improve the current individual in the process of evolution. For example, Genetic Powell Learning (GPL) is a combination of real-coded GA and Powell's method [18]. (3) (E < T > E)-type, in which the conventional operator is directly used as an evolutionary operator. For example, simplex-GA [18] and simplex coding genetic algorithm [8] are hybridizations of real-coded GA and simplex method.

The above three ways of hybridization do work to improve the performance of EA. But they are just some combinations of existing operators. Disharmony between evolutionary and conventional operators might arise. The hybrid algorithm needs a tune-up to fit the combination. To this end, we try modifying the involved operators essentially to take advantage of their specialties. This method is referred to as **(< ET >)-type** hybridization, where < ET > means evolutionary and conventional operators are closely combined. In this new way of hybridization, we have attained a hybrid EA named low-dimensional simplex evolution (LDSE) [12] (detailed abstract of LDSE was presented at SNPD 2007). LDSE uses simplex operators conditionally and selectively with essential modifications. It has shown its advantage against differential evolution with random localization (DERL) [10], which is an improved version of DE [21] proposed by Kaelo and Ali. Basic form of LDSE (basic LDSE) has been successfully applied to some practical problems [13,23].

However, basic LDSE needs a large population to ensure its convergence. Obviously, the large population-size could slow down LDSE's convergence speed. There are at least three reasons why basic LDSE needs a large population: (1) Its reproduction operators (including reflection, contraction and struggle) are all linear. Therefore, a large population is required to maintain its population diversity. (2) The simplex method might be failed in high-dimensional space if the population size is not large enough. (3) Its local search ability will be dramatically reduced if it uses a small population.

In this paper, we will introduce three techniques to improve LDSE's performance. To reduce its required population size, we try to (1) use nonlinear operators to replace the linear ones, (2) implement simplex operations in low-dimensional spaces, and (3) enhance LDSE's local search ability. Specifically, the three techniques are called low-dimensional reproduction (LDR), normal struggle (NS) and variable dimension (VD). LDR keeps some (randomly chosen) components fixed to those of the current individual. As a result, the offspring individuals are capable of escaping from the hyperplane determined by their parents, and the elitism information is partially preserved. This can help to maintain the diversity of population. Numerical results show that LDSE with LDR can converge with less population size than LDSE without LDR. NS tries to enhance LDSE's local search capability by allowing unlucky individual search around the best vertex of m-simplex. VD tries to draw lessons from recent failure by making further exploitation on its most promising sub-facet. Numerical experiments indicate the three techniques can work together, and LDSE with the three techniques (modified LDSE) outperforms basic LDSE considerably.

Comparison of modified LDSE and DERL (an improved version of DE) is also carried out. The result shows that modified LDSE can find the global optimum with fewer numbers of evaluations (nfe) in most cases, and it can also solve some problems which DERL cannot solve within a given number of nfe.

The convergence properties of LDSE and its modifications are also analysed in this paper. It shows that basic LDSE might fail to converge but modified LDSE with the above three techniques will converge for any initial population. To evaluate the convergence speed of modified LDSE, an estimation of its first passage time (of reaching the global minimum) is provided.

The rest of this paper is organized as follows. Basic LDSE is briefly reviewed in Section 2. The three techniques are introduced in Section 3. Convergence analyses are discussed in Section 4. Numerical results are provided in Section 5. Concluding remarks are drawn in Section 6.

2. Basic form of low-dimensional simplex evolution

LDSE [12] is a real-coded EA for GO of continuous variables. Similar to other EAs, LDSE is also population-set-based and stochastic. The individuals survive by the rule of natural selection (i.e. survival of the fittest). Different from other EAs, LDSE generates new individuals with low-dimensional simplex operators selectively and conditionally, and the individuals are updated in the framework of try-try-struggle. Let the population size be N, for each individual $\boldsymbol{X}_i(t)$ ($i = 1, 2, \ldots, N$) in current population $\vec{\boldsymbol{X}}(t)$, LDSE selects $m+1$ points (vectors or individuals) randomly to form an m-simplex, where m is usually much less than the problem dimension n. Then, LDSE will find the best $\boldsymbol{X}_\mathrm{b}$ and the worst $\boldsymbol{X}_\mathrm{w}$ among the $m+1$ points. In LDSE, an individual $\boldsymbol{X}_i(t)$ has two chances to improve at every generation t. The first chance is provided by reflection. The reflection point $\boldsymbol{X}_r$ will replace $\boldsymbol{X}_i(t)$ if it is better. Thus, the ith individual $\boldsymbol{X}_i(t)$ is improved. Otherwise, the second chance, contraction, will be carried out. Similarly, the contraction point $\boldsymbol{X}_c$ will replace $\boldsymbol{X}_i(t)$ if it is better. The reflection point $\boldsymbol{X}_r$ is generated by reflecting the worst point $\boldsymbol{X}_\mathrm{w}$ through the centroid of the other m points $\bar{\boldsymbol{X}}$, i.e. $\boldsymbol{X}_r = \bar{\boldsymbol{X}} + \alpha \cdot (\bar{\boldsymbol{X}} - \boldsymbol{X}_\mathrm{w})$, and the contraction point $\boldsymbol{X}_\mathrm{c}$ complies with the rule $\boldsymbol{X}_c = \bar{\boldsymbol{X}} + \beta \cdot (\boldsymbol{X}_\mathrm{w} - \bar{\boldsymbol{X}})$, where α and β are scaling factors of preset positive numbers. If the i-th individual $\boldsymbol{X}_i(t)$ has lost the above two chances, and it cannot receive its average profit, that is, its function value is greater than or equal to the average value of the current population (i.e. $f(\boldsymbol{X}_i(t)) \geq (1/N) \sum_i f(\boldsymbol{X}_i(t))$), it will take its last struggle to step towards the best individual $\boldsymbol{X}_\mathrm{b}$ (i.e. the struggle point $\boldsymbol{X}_s = \boldsymbol{X}_i + 0.618 \cdot (\boldsymbol{X}_\mathrm{b} - \boldsymbol{X}_i)$) or away from the worst individual $\boldsymbol{X}_\mathrm{w}$ (i.e. $\boldsymbol{X}_s = \boldsymbol{X}_i + 0.382 \cdot (\boldsymbol{X}_i - \boldsymbol{X}_\mathrm{w})$). LDSE's hybridization way is ($<$ ET $>$)-type. It comprises the ideas of Nelder-Mead, ES and elite preservation. However, the individuals are generated in low dimensional spaces and updated in a framework of try-try-struggle.

3. Three techniques introduced to improve basic LDSE

3.1 *Low-dimensional reproduction*

From Section 2, we can see that the three reproduction operators, including reflection, contraction and struggle, are all linear. As a result, new individuals cannot escape from the hyperplane determined by their parent individuals (see Figure 1(a)). Therefore, basic LDSE needs a large population to ensure its convergence. This might slow down its convergence speed. Therefore, we want to change the linear operators to nonlinear ones.

Subspace technique is successfully used in linear algebra. For example, Krylov subspace methods are regarded as one of the 10 most-important classes of numerical methods [7]. Another fact

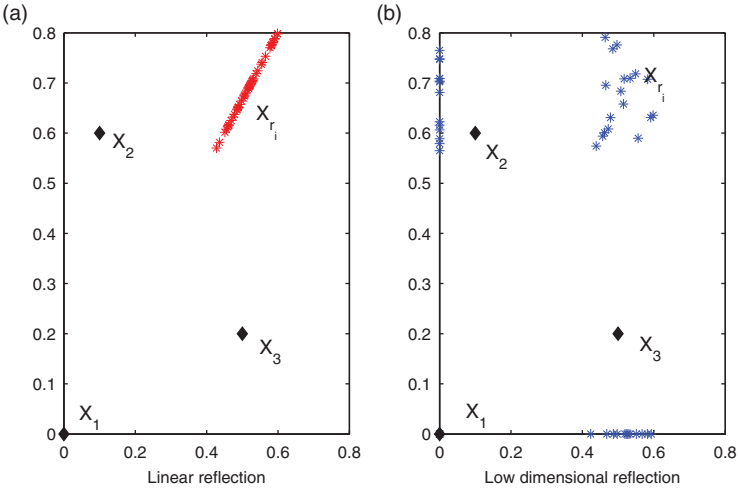

Figure 1. Possible offsprings generated by reflection with and without LDR technique in case $m = 2$ and $n = 2$, where X_1, X_2, and X_3 are parents, and X_1 and X_3 are the worst and the best vertex, respectively. Fifty possible offsprings are shown, only one is labelled as X_{r_i}.

is that the simplex method is quite efficient for low-dimensional problems. These facts motivate us to use the simplex method in low-dimensional spaces.

Elitism selection is a widely used technique in EAs, in which the elite individual is preserved for the next generation. It has been successfully used to ensure their global convergence and accelerate the convergence speed. Therefore, we want to combine elitism selection with reproduction operators in LDSE.

Now the question is how to implement the above three ideas with one modification. In fact, this is not a difficult task. We have implemented it in a simple way: (1) keep some (randomly chosen) components of a local best vector (regarded as an elite individual here) for its offspring in the reproduction process, and (2) let the original reproduction operators act only on the rest components.

In this way, the reproduction operator is no longer linear. The problems caused by the linearity are avoided. From Figure 1(b), we can see that the nonlinear operator is more suitable for maintaining the diversity of the population. Now that the simplex operators act only on a part of components, the searching process (a reflection or contraction) is actually limited to a subspace of $\mathbb{R}^n$. Some components of the elite individual are kept for its offspring. So the elite individual is partially preserved. This is an incomplete elitism-selection process.

The above technique is referred to as LDR. The reflection in basic LDSE is

$$\boldsymbol{X}_r = \bar{\boldsymbol{X}} + \alpha \cdot (\bar{\boldsymbol{X}} - \boldsymbol{X}_{\mathrm{w}}).$$

The reflection with LDR technique (also referred to as low-dimensional reflection, see Figure 1(b)) becomes

$$x_{r,j} = \begin{cases} x_{i,j} & \text{if } p_r < p_{\mathrm{a}}, \\ \bar{x}_j + \alpha \cdot (\bar{x}_j - x_{w,j}) & \text{else,} \end{cases} \qquad j = 1, 2, \ldots, n,$$

where p_r is a random variable with uniform distribution generated from [0,1]; $x_{r,j}$, $x_{i,j}$, $\bar{x}_j$ and $x_{w,j}$ are the jth component of the reflection point $\boldsymbol{X}_r$, the current individual $\boldsymbol{X}_i$, the centroid $\bar{\boldsymbol{X}}$ and the worst vertex $\boldsymbol{X}_w$, respectively; p_{a} is a preset number called adsorption probability, and $0 \le p_{\mathrm{a}} \le 1$.

Low-dimensional contraction with LDR technique is defined similarly. The contraction point $\boldsymbol{X}_c$ is generated as follows.

$$x_{c,j} = \begin{cases} x_{i,j} & \text{if } p_r < p_a, \\ \bar{x}_j + \beta \cdot (x_{w,j} - \bar{x}_j) & \text{else,} \end{cases} \qquad j = 1, 2, \ldots, n.$$

LDR can help offsprings escape from the hyperplane determined by their parents. Meanwhile, it can prevent them from getting too decentralized and help to search mainly along a series of orthogonal directions (coordinate axes) (see Figure 1(b)). LDR has something in common with elitism selection if the current individual is considered as an elite. Both LDR and elitism selection preserve something of the elite for its offsprings. However, in LDR, the preserved object is not the elite itself, but some components of it. LDSE with low-dimensional reflection and low-dimensional contraction techniques is referred to as LDR-LDSE.

3.2 *Normal struggle*

As previously mentioned, the struggle operator in basic LDSE is linear. This is not good for maintaining the diversity of the population, because the search space is limited to 1-dimension (two line segments, see Figure 2(a)). So we consider replacing the linear struggle operator with a nonlinear one. The struggled point is defined as a normal disturbance of the best vertex of m-simplex (see Figure 2(b)), i.e. $\boldsymbol{X}_s = \boldsymbol{X}_b(t) + \boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, where $\boldsymbol{X}_b(t)$ is the best vertex of m-simplex, and $\boldsymbol{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ is a random vector of normal distribution with mean $\boldsymbol{\mu}$ and standard variance $\boldsymbol{\sigma}^2$. This nonlinear struggle operator is referred to as normal struggle (NS).

Obviously, NS can enhance local search around local best individual (the best vertex of m-simplex). NS can also increase the diversity of the population. In this paper, NS is always used in conjunction with LDR. The LDSE with NS and LDR will be referred to as NS-LDSE.

The idea of NS is borrowed from evolution strategy (ES) [17]. However, they are not exactly the same. First, NS-LDSE uses low-dimensional normal distributions. Second, NS is based on the best vertex of m-simplex, not the current individual (in ES). Third, NS will not take effect if reflection or contraction has successfully improved the current individual.

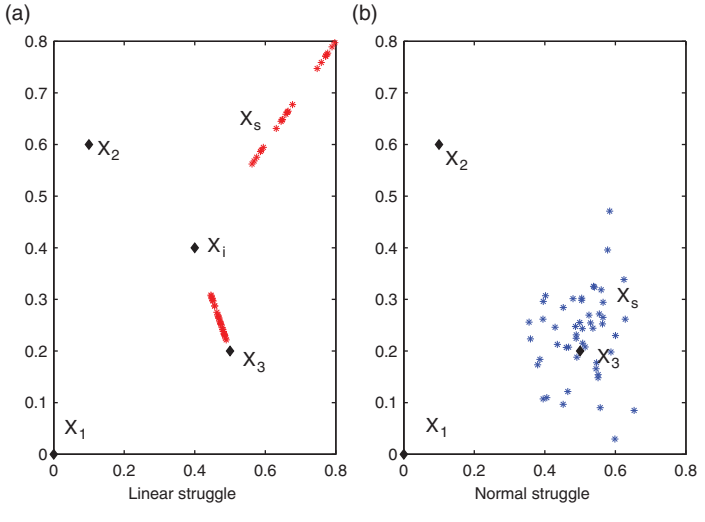

Figure 2. Possible offsprings generated by linear struggle and normal struggle in case $m = 2$ and $n = 2$, where X_1, X_2, X_3, and X_i are parents, and X_1 and X_3 are the worst and the best vertex, respectively. Fifty possible offsprings are shown, only one is labelled as X_s.

3.3 *Variable dimension*

The essential difference between LDSE and other EAs is that LDSE is based on realtime-generated m-simplex to find a most promising direction and an adapted step-length. In basic LDSE, each m-simplex is used only once, and then it will be discarded. Clearly, this might be a waste, since we have just got the latest information about the current m-simplex. To make use of this information, we introduce a VD technique. VD tries to make further exploitation around the current simplex (applying lower-dimensional simplex operators) if the m-simplex operators (including reflection and contraction) fail to improve the current individual.

Note that m-simplex has many sub-facets. Specifically, it has C_{m+1}^{k+1} sub-facets with dimension k (also referred to as k-facets), where $k = 1, 2, \ldots, m-1$. Obviously, it is very important to choose an appropriate sub-facet on which the simplex operation is conducted. We will introduce a new concept, the most promising sub-facet, before proceeding. The most promising sub-facet is defined such that the simplex operators on it will most likely lead to a descent direction.

DEFINITION 3.1 *A k-facet is said to be the most promising among all k-facets if it is selected in the following steps.*

Step 1: *Let D_f be the set of k-facets defined as D_f = k-facet in m-simplex | The k-facet has the largest function–value difference between its worst and the second worst vertexes*}. *If there is only one k-facet in set D_f, it will be selected as the most promising sub-facet, and the selection process stops. Otherwise, continue to the next step.*

Step 2: *Let V_f be the set of k-facets defined as V_f = k-facet in D_f | The k-facet has the smallest function–value variance of all its vertexes except the worst one*}. *If there is only one k-facet in set V_f, then it will be selected as the most promising sub-facet, and the selection process stops. Otherwise, continue to the next step.*

Step 3: *Let V_d be the set of k-facets defined as V_d = k-facet in V_f | The k-facet has the smallest distance variance of its worst vertex to the rest vertexes*}. *If there is only one k-facet in set V_d, it will be selected as the most promising sub-facet, and the selection process stops. Otherwise, select the first one as the most promising sub-facet, and the selection process stops.*

Remark From the above definition, we can see that it might be a multi-level optimization process to find the most promising sub-facet. If there are several sub-facets with the same function–value difference in step 1, all of them will be candidates of the most promising sub-facet. Similarly, if there are more than one sub-facets in set D_f with the same function–value variance, any of them will be a candidate of the most promising sub-facet. Finally, if there are still several sub-facets in set V_f with the same distance variance, then only the first one will be selected as the most promising sub-facet. Therefore, there will be one and only one k-facet selected as the most promising one at each iteration. It is common that some sub-facets share the same edge. In this case, the set D_f consists of several sub-facets. The selection process is demonstrated in Examples 3.2 and 3.3 as follows.

LDSE with VD technique detects a most promising sub-facet of the current m-simplex first, and then applies lower-dimensional simplex operators (including reflection and contraction) on it. This process will be repeated if the lower-dimensional simplex operators are failed again. In other words, k-simplex operators will be conducted recursively if all the previous simplex operators with higher dimensions are failed, where $k = m-1, m-2, \ldots, 2$ (see Example 3.2). Note that the total number of k-facets of an m-simplex is C_{m+1}^{k+1}. However, here the k-simplex operators are carried out only on the most promising k-facet. VD technique can help to draw lessons from recent failure and make the most of the latest local information (provided by the current m-simplex). In this paper, VD is also always used in conjunction with LDR. LDSE with VD and LDR techniques is denoted by VD-LDSE.

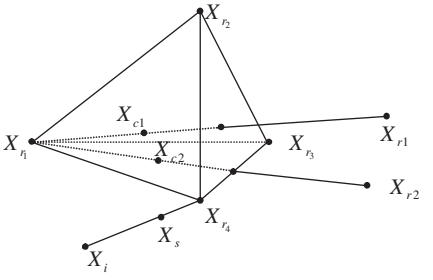

Figure 3. A possible way of applying simplex operators in case $m = 3$ and $n = 3$: $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$ and $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$. Five parents include four vertices $\boldsymbol{X}_{r1}, \boldsymbol{X}_{r2}, \boldsymbol{X}_{r3}$ and $\boldsymbol{X}_{r4}$ and the current individual $\boldsymbol{X}_i$. Possible offsprings are $\boldsymbol{X}_{r1}$, $\boldsymbol{X}_{c1}$, $\boldsymbol{X}_{r2}$, $\boldsymbol{X}_{c2}$ and $\boldsymbol{X}_s$, which will be generated in turn if and only if the previous offspring failed to improve the current individual.

Note that VD technique will not be activated if the dimension of the original simplex satisfies $m = 2$. In this case, VD-LDSE and LDR-LDSE are the same.

Example 3.2 Consider VD-LDSE in case $m = 3$, and let $f(\boldsymbol{X}_{r_1}) \geq f(\boldsymbol{X}_{r_2}) \geq f(\boldsymbol{X}_{r_3}) \geq f(\boldsymbol{X}_{r_4})$. Each individual will have four chances to improve, namely, 3-simplex reflection, 3-simplex contraction, 2-simplex reflection and 2-simplex contraction. 3-simplex operators are done in the way of $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$. The two-dimensional sub-facet (2-simplex) on which the 2-simplex operators are done is chosen in the following way:

If $f(\boldsymbol{X}_{r_3}) - f(\boldsymbol{X}_{r_2}) < f(\boldsymbol{X}_{r_4}) - f(\boldsymbol{X}_{r_3})$, $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}$, else if $\text{abs}(|\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_3}| - |\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_4}|) < \text{abs}(|\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}| - |\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}|)$, $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$, else $\boldsymbol{X}_{r_2} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$.

There are three possible ways of using the 2-simplex operators in Example 3.2. Figure 3 shows only one possible way in $\mathbb{R}^3$. Note that there are four two-dimensional sub-facets in Example 3.2, namely, $\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}$, $\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$, $\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$ and $\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}$. However, 2-simplex operators should not be carried out on facet $\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}$. The reasons are as follows. For the sub-facet $\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}$, we have $f(\boldsymbol{X}_{r_4}) - f(\boldsymbol{X}_{r_2}) \geq f(\boldsymbol{X}_{r_4}) - f(\boldsymbol{X}_{r_3})$ and $f(\boldsymbol{X}_{r_4}) - f(\boldsymbol{X}_{r_2}) \geq f(\boldsymbol{X}_{r_3}) - f(\boldsymbol{X}_{r_2})$. So the angle between $\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}$ and the contour surface (through $\boldsymbol{X}_{r_4}$) is likely greater than that of $\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$. Similarly, the angle between $\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}$ and the contour surface (through $\boldsymbol{X}_{r_2}$) is likely greater than that of $\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}$. This means that the chance of getting a decent direction is little if we choose the sub-facet $\boldsymbol{X}_{r_1}\boldsymbol{X}_{r_2}\boldsymbol{X}_{r_4}$ to carry out the 2-simplex operators.

Example 3.3 Consider VD-LDSE in case $m = 4$, and let $f(\boldsymbol{X}_{r_1}) \geq f(\boldsymbol{X}_{r_2}) \geq f(\boldsymbol{X}_{r_3}) \geq f(\boldsymbol{X}_{r_4}) \geq f(\boldsymbol{X}_{r_5})$. Each individual will have six chances to improve, namely, two 4-simplex operators, two 3-simplex operators, and two 2-simplex operators.

The 4-simplex operators are done in the way of $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}\boldsymbol{X}_{r_5}$.

The three-dimensional operators are done in three possible ways: (1) $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$ or (2) $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}\boldsymbol{X}_{r_5}$ or (3) $\boldsymbol{X}_{r_2} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}\boldsymbol{X}_{r_5}$. If $\text{var}(f(\boldsymbol{X}_{r_2}), f(\boldsymbol{X}_{r_3}), f(\boldsymbol{X}_{r_4})) < var(f(\boldsymbol{X}_{r_3}), f(\boldsymbol{X}_{r_4}), f(\boldsymbol{X}_{r_5}))$, choose (1); else if $var(d_{13}, d_{14}, d_{15}) < var(d_{23}, d_{24}, d_{25})$, then choose (2); otherwise, choose (3).

The two-dimensional operators are done in six possible ways: (1) $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_2}\boldsymbol{X}_{r_3}$ or (2) $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$ or (3) $\boldsymbol{X}_{r_1} \rightarrow \boldsymbol{X}_{r_4}\boldsymbol{X}_{r_5}$ or (4) $\boldsymbol{X}_{r_2} \rightarrow \boldsymbol{X}_{r_3}\boldsymbol{X}_{r_4}$ or (5) $\boldsymbol{X}_{r_2} \rightarrow \boldsymbol{X}_{r_4}\boldsymbol{X}_{r_5}$ or (6) $\boldsymbol{X}_{r_3} \rightarrow \boldsymbol{X}_{r_4}\boldsymbol{X}_{r_5}$. First, find the minimum of $(f(\boldsymbol{X}_{r_3}) - f(\boldsymbol{X}_{r_2}))$, $(f(\boldsymbol{X}_{r_4}) - f(\boldsymbol{X}_{r_3}))$, and $(f(\boldsymbol{X}_{r_5}) - f(\boldsymbol{X}_{r_4}))$. If $(f(\boldsymbol{X}_{r_3}) - f(\boldsymbol{X}_{r_2}))$ is the minimum, choose (1). If $(f(\boldsymbol{X}_{r_4}) - f(\boldsymbol{X}_{r_3}))$ is the minimum, choose (2) or (4); furthermore, if $|d_{13} - d_{14}| < |d_{23} - d_{24}|$, then choose (2), otherwise choose (4). If $(f(\boldsymbol{X}_{r_5}) - f(\boldsymbol{X}_{r_4}))$ is the minimum, choose (3), or (5), or (6); furthermore, find $i^* = \arg\min_{i \in \{1,2,3\}} |d_{i4} - d_{i5}|$. If $i^* = 1$, choose (3), else if $i^* = 2$, choose (5), otherwise ($i^* = 3$), choose (6). Here, $d_{ij} = |\boldsymbol{X}_{r_i}\boldsymbol{X}_{r_j}|$ denotes the distance between $\boldsymbol{X}_{r_i}$ and $\boldsymbol{X}_{r_j}$.

From Example 3.3, we can see that there are at least 18 possible ways (three possible three-dimensional operators times six possible two-dimensional operators) of generating a new individual in case the simplex dimension $m = 4$.

As can be seen from Examples 3.2 and 3.3, identifying the most promising sub-facet is a key step in LDSE with VD technique. By Definition 3.1, the identification method in general cases is as follows. Let $\boldsymbol{X}_{\mathrm{w}} \to \boldsymbol{X}_{l_1}\boldsymbol{X}_{l_2}\cdots\boldsymbol{X}_{l_k}$ denote a k-simplex operator on the most promising sub-facet, where $f(\boldsymbol{X}_{\mathrm{w}}) > f(\boldsymbol{X}_{l_1}) > f(\boldsymbol{X}_{l_2}) > \cdots > f(\boldsymbol{X}_{l_k})$, then the most promising means that (1) the difference $f(\boldsymbol{X}_{\mathrm{w}}) - f(\boldsymbol{X}_{l_1})$ should be as large as possible, (2) the variance $\mathrm{var}(f(\boldsymbol{X}_{l_1}), f(\boldsymbol{X}_{l_2}), \ldots, f(\boldsymbol{X}_{l_k}))$ should be as small as possible, and (3) the variance $\mathrm{var}(d_{\boldsymbol{X}_{\mathrm{w}}\boldsymbol{X}_{l_1}}, d_{\boldsymbol{X}_{\mathrm{w}}\boldsymbol{X}_{l_2}}, \ldots, d_{\boldsymbol{X}_{\mathrm{w}}\boldsymbol{X}_{l_k}})$ should be as small as possible.

It is worth pointing out that the evolution process of basic LDSE, as well as LDR-LDSE and NS-LDSE should be non-generational, that is to say, an individual will be used as a parent individual to generate offsprings as soon as it is generated. In fact, non-generational evolution can help to save storage space and sometimes can accelerate the convergence speed as well. However, VD-LDSE should be generational if $m \geq 3$. In fact, the matrix of individual-distances is needed to find the most promising sub-facet (as shown in general case (3)). If the individuals change frequently, then the matrix of individual-distances needs to update at the same time. As a result, the identification of the most promising sub-facet might become very expensive, especially for high-dimensional problems.

3.4 *Procedure of modified LDSE*

In this subsection, we will integrate the above three techniques together into basic LDSE. The resulted algorithm is referred to as modified LDSE. Its flowchart is shown in Figure 4 and its procedure can be described as follows.

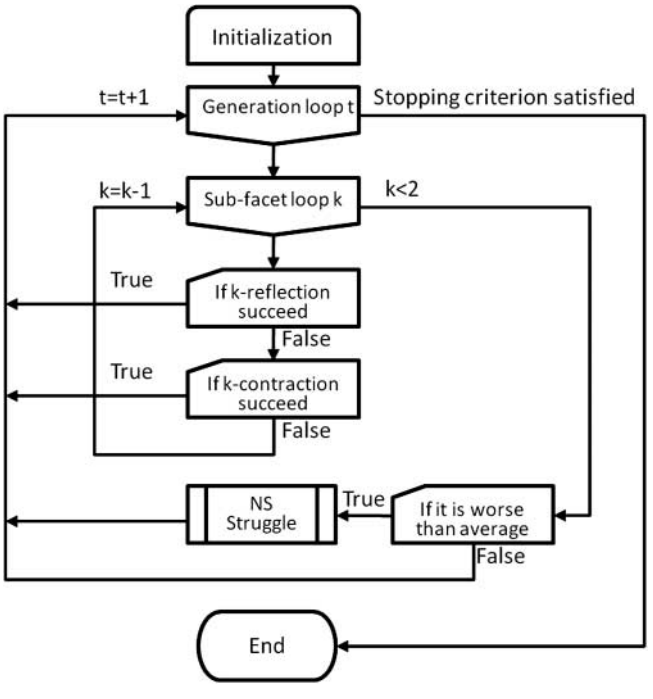

Figure 4. Flow chart of modified LDSE.

Procedure of modified LDSE:

Step 1: Initialize: Input population size N, initial bounds l and u, simplex dimension m, adsorption probability p_a, scaling factors α and β, struggle scale σ. Set the current generation $t=0$; And initialize population $\vec{X}(0)=\{X_1(0), X_2(0), \ldots, X_N(0)\}$, where $X_i(0)\in\mathbb{R}^n$.

Step 2: Evaluate population: For each individual in the current population $\vec{X}(t)$, compute $f(X_i(t))$; set the current position $i=1$.

Step 3: Update population: If the current position $i\le N$, then set success mark suc = False, and do step 3.1–3.3.

(3.1) Construct simplex: Randomly choose $m+1$ mutually different individuals X_{r_k}, $k=1,2,\ldots,m+1$ from current population, find their best vertex X_b; Set the dimension of sub-facet $k=m$.

(3.2) If the dimension of sub-facet $k>1$, then do steps 3.2.1–3.2.4. **/* VD technique */**

(3.2.1) Identify sub-facet: Find the most promising k-facet of the m-simplex and its worst vertex X_w, and calculate its centroid $\bar{X}$ except X_w.

(3.2.2) Try reflection: Generate a reflection point $X_r(t)=(x_{r,1}(t), x_{r,2}(t), \ldots, x_{r,n}(t))$ as follows.

for ($j=1$; $j\le n$; $j++$){
  if $rand < p_a$ **/* LDR technique */**
    $x_{r,j}(t)=x_{i,j}(t)$
  else
    $x_{r,j}(t)=\bar{x}_j+\alpha\cdot(\bar{x}_j-x_{w,j})$
}

If $f(X_r)<f(X_i(t))$, then $X_i(t+1)=X_r$,
set the current position $i=i+1$, and then return to step 3.

(3.2.3) Try contraction: Generate a contraction point $X_c(t)=(x_{c,1}(t), x_{c,2}(t), \ldots, x_{c,n}(t))$ as follows.

for (j=1; $j\le n$; j++){
  if $rand < p_a$, **/* LDR technique */**
    $x_{c,j}(t)=x_{i,j}(t)$
  else
    $x_{c,j}(t)=\bar{x}_j+\beta\cdot(x_{w,j}-\bar{x}_j)$
}

If $f(X_c)<f(X_i(t))$, then $X_i(t+1)=X_c$,
set the current position $i=i+1$, and then return to step 3.

(3.2.4) Check point: If $k>2$, Set the dimension of sub-facet $k=k-1$, and then return to step 3.2.

(3.3) Struggle: If the current individual $X_i(t)$ is worse than average, i.e. $f(X_i(t))\ge(1/N)\sum_{k=1}^{N} f(X_k(t))$, then generate a struggle point $X_s=(x_{s,1}, x_{s,2}, \ldots, x_{s,n})$ as follows.

for (j=1; $j\le n$; j++){
  if $rand < p_a$ **/* NS and LDR techniques */**
    $x_{s,j}=x_{b,j}$
  else
    $x_{s,j}=x_{b,j}+N(0,\sigma^2)$.
}

Let $X_i(t+1)=X_s$, set the current position $i=i+1$, and then return to step 3.

Step 4: Check point: If some stopping criterion is satisfied, output the best-so-far individual X^* and its function value $f(X^*)$; otherwise, set the current generation $t=t+1$, set the current position $i=1$ and then return to step 3.

From the above procedure and Figure 4, whether the k-reflection (or k-contraction) is successful or not depends on the function values of the new generated individual and the current one. The current individual will hibernate until next generation once it gets improved. It has $2^*(m-1)$ chances to improve. It will take its last struggle only if it is worse than average and has lost all of the chances.

The initial population is usually randomly scattered within the given initial bounds. Of course you can specify some or all of them to expect a faster convergence if you have some previous information. Stopping criteria are easy to set. You may specify a maximum number of generations/function-evaluations, a time limit, a fitness goal, or a number of stall generations to determine when to terminate the searching process.

4. Convergence analysis of LDSEs

Although empirical evidences indicate that LDSE and its modifications can find global optimal solutions in most cases, there is no theoretical result which can be used directly to analyze its convergence. In this section, we analyse the convergence properties of basic LDSE and its modifications by homogeneous finite Markov chains. We also present an upper bound of its first passage time (of reaching the global minimum) to evaluate the convergence speed of modified LDSE.

4.1 *Convergence properties of LDSEs*

Markov chain is often used to analyse the convergence properties of EAs [3,2,16,20]. A most popular Markov chain analysis of genetic algorithms is made by Rudolph [19]. He analysed the convergence properties of canonical genetic algorithms (CGAs). Next we will show that the evolutionary process of LDSE, similar to that of CGAs, could be also described as a Markov chain. However, no result could be used directly to analyse the convergence properties of LDSE, because LDSE is real-coded and its transition matrix is quite different from that of other EAs. This makes the analysis of its convergence properties necessary.

Consider the GO problem of the following form:

$$\min_{\boldsymbol{x}\in\boldsymbol{\Omega}\subset\mathbb{R}^n} f(\boldsymbol{x}).$$

Suppose that it has finite global optimal value f^* with optimal solution $\boldsymbol{x}^*$. In finite precision, the search space $\boldsymbol{\Omega}$ is a finite point-set in $\mathbb{R}^n$, and each point $\boldsymbol{x}_i$ corresponds to an individual $\boldsymbol{X}_i(t)$ in current population $\vec{\boldsymbol{X}}(t)$. Let the population size be N, then each population $\vec{\boldsymbol{X}}(t)$ maps to an N-point-set and also maps to a state of the evolutionary process S_k (see Figure 5). Suppose that there are L points in the search space Ω, and the N-point-set allow to duplicate elements, then the total number of possible N-point-sets will be L^N. Let the state space be $\Gamma=\{S_1, S_2, \ldots, S_M\}$, where M is the cardinality number, i.e. $|\Gamma|=M$. As shown in Figure 5, all N-point-sets are mapped to all possible populations and all possible populations are mapped to all states. So there exists a one-to-one map between the space of N-point-set and the state space. If the population allows to duplicate individuals, we can get $M=L^N$. For the convenience of representation, sometimes a population $\vec{\boldsymbol{X}}(t)$ is directly regarded as a state of the evolutionary process.

DEFINITION 4.1 *A state S^* is said to be optimal, if its corresponding N-point-set contains an optimal point $\boldsymbol{x}^*$ such that $\forall \boldsymbol{x}\in\Omega, f(\boldsymbol{x}^*)\le f(\boldsymbol{x})$, and its corresponding population is called an optimal population, denoted by $\vec{\boldsymbol{X}}^*$. The set of all optimal states is called the optimal state set, denoted by Γ^*.*

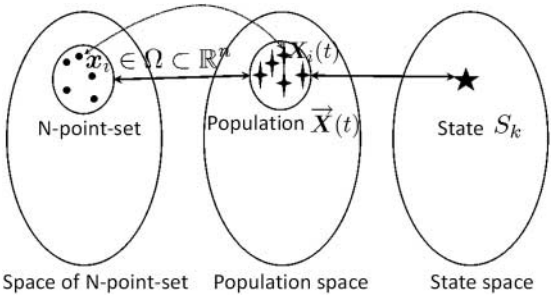

Figure 5. Relationship of three spaces.

Based on the above definition, the state space Γ can be divided into two parts, i.e. $\Gamma = \Gamma_1 \cup \Gamma_2$, where $\Gamma_1 = \Gamma^*, \Gamma_2 = \Gamma \backslash \Gamma^*$.

Suppose that a population $\vec{\boldsymbol{X}}$ is in state S_i with probability q_i, $i = \{1, 2, \ldots, M\}$, then we have

(1) $0 \le q_i \le 1$.
(2) $\sum_{i=1}^{M} q_i = 1$.
(3) The state distribution of a population $\vec{\boldsymbol{X}}$ could be described by a probability vector $\boldsymbol{q} = (q_1, q_2, \ldots, q_M)$.

Let $\boldsymbol{q}^{(t)} = (q_1^{(t)}, q_2^{(t)}, \ldots, q_M^{(t)})$ denote the distribution vector at generation t, then $\boldsymbol{q}^{(0)}$ represents the distribution of initial population $\vec{\boldsymbol{X}}(0)$. Suppose that the transition matrix P exists, and the distribution vector at generation t is $\boldsymbol{q}^{(t)}$, then the distribution vector at generation $t+1$ would be $\boldsymbol{q}^{(t+1)} = \boldsymbol{q}^{(t)} P$, where P is a $M \times M$ stochastic matrix such that $0 \le p_{ij} \le 1$ and $\sum_{j=1}^{M} p_{ij} = 1$, $i = \{1, 2, \ldots, M\}$. The sequence of the distribution vectors $\{\boldsymbol{q}^{(t)}\}_{t=0}^{\infty}$ forms a Markov chain. Accordingly, the sequence of the population $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ is also called a Markov chain. If there exists a distribution vector $\boldsymbol{q}^{\infty}$, such that $\boldsymbol{q}^{(t)} \to \boldsymbol{q}^{\infty}$, as $t \to \infty$, then its corresponding Markov chain is said to be convergent, and $\boldsymbol{q}^{\infty}$ is called the limit distribution of the Markov chain, or the stationary distribution of the transition matrix P. In this case, $P^t \to P^{\infty}$, $\boldsymbol{q}^{\infty} = \boldsymbol{q}^{(0)} P^{\infty}$, and the Markov sequence $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ is said to be convergent.

Remark It is worth noticing that the convergence of the population sequence $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ generated by LDSE, or its state distribution $\{\boldsymbol{q}^{(t)}\}_{t=0}^{\infty}$ is not equivalent to the convergence of LDSE. In fact, the convergence of $\{\boldsymbol{q}^{(t)}\}_{t=0}^{\infty}$ is only a necessary condition for LDSE to be convergent. If there exists a component $q_i^{\infty} > 0$ in the limit distribution vector $\boldsymbol{q}^{\infty}$, and its corresponding state is not global optimal, then the LDSE could not be guaranteed to converge to the global minimizer, even in infinite computing time. To ensure that the LDSE converges to the global minimum, the limit distribution vector $\boldsymbol{q}^{\infty}$ must satisfy that the sum of the components corresponding to the optimal states in it be equal to 1, and all other components equal to 0, that is, $\forall i \in \{1, 2, \ldots, M\}$, if $S_i \notin \Gamma^*$, then $q_i^{\infty} = 0$, and $\sum_{S_i \in \Gamma^*} q_i^{\infty} = 1$ for all states in Γ^*.

Now we conclude the above discussions as the following lemma:

LEMMA 4.2 *Let $\vec{\boldsymbol{X}}(t)$ $(t = 0, 1, 2, \ldots)$ be the population sequence generated by an EA, Γ^* denote its optimal state set, $\boldsymbol{q}^{(t)}$ be the state distribution of population $\vec{\boldsymbol{X}}(t)$. If its limit distribution $\boldsymbol{q}^{\infty}$ satisfies*

$$q_i^{\infty} = \begin{cases} 0 & \text{if } S_i \notin \Gamma^*, \\ q_i^{\infty} \ge 0 & \text{if } S_i \in \Gamma^*, \end{cases} \quad \text{for all } i = 1, 2, \ldots, M$$

and $\sum_{S_i \in \Gamma^*} q_i^{\infty} = 1$, then the EA converges to the global minimum.

THEOREM 4.3 *In finite precision, the population sequence* $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ *generated by LDSE is a homogeneous finite Markov chain.*

Proof First, the new population $\vec{\boldsymbol{X}}(t+1)$ depends only on the current population $\vec{\boldsymbol{X}}(t)$ and has nothing to do with the former generations $\vec{\boldsymbol{X}}(\tau)$, $\tau < t$. Thus, the population sequence $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ generated by LDSE is a Markov chain.

Second, the evolutionary operators in LDSE (either basic LDSE or modified LDSE) are all time independent. As a result, the Markov chain generated by LDSE is homogeneous.

Third, as discussed above, the number of all possible populations is finite in finite precision (although LDSE is real-coded), i.e. the state space is finite. Consequently, the Markov chain generated by LDSE is finite. ■

Note that the population sequence $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ generated by CGA is a homogeneous, irreducible, aperiodic and finite Markov chain. But for LDSE, the Markov chain is neither irreducible nor aperiodic. In fact, we have the following results.

THEOREM 4.4 *For the population Markov chain* $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ *generated by LDSE, the following state sets are closed:* (1) *the optimal state set* Γ^**;* (2) (*for basic LDSE*) *the state set corresponding to some populations distributed in the same subspace of* $\mathbb{R}^n$; (3) (*for basic LDSE*) *the state set corresponding to uniform populations* (*in which all individuals are same*).

Proof (1) If the current population $\vec{\boldsymbol{X}}(t)$ contains an optimal individual $\boldsymbol{X}^*$, then all the latter populations $\vec{\boldsymbol{X}}(\tau)$, $\tau > t$ will always contain optimal individual(s) during the evolutionary process of LDSE. In fact, reflection point $\boldsymbol{X}_r(t)$ (or contraction point $\boldsymbol{X}_c(t)$) replaces the current individual $\boldsymbol{X}_i(t)$ if and only if it is better than $\boldsymbol{X}_i(t)$. Therefore, reflection and contraction operators will not remove the optimal individual $\boldsymbol{X}^*$ from the population $\vec{\boldsymbol{X}}(t)$. The last struggle operator might remove an optimal individual $\boldsymbol{X}^*$ in $\vec{\boldsymbol{X}}(t)$, but the population $\vec{\boldsymbol{X}}(t)$ would still contain some other optimal individuals $\hat{\boldsymbol{X}}^*$. In fact, the last struggle will take place only if the current individual $\boldsymbol{X}_i(t)$ is not better than the average of the population, i.e. $f(\boldsymbol{X}_i(t)) \geq \bar{f} = (1/N)\sum_j f(\boldsymbol{X}_j(t))$. Meanwhile, for an optimal individual $\boldsymbol{X}^*$, we have $f(\boldsymbol{x}^*) \leq \bar{f}$. Consequently, an optimal individual takes its last struggle if and only if $f(\boldsymbol{x}^*) = \bar{f}$, which implies all individuals in the current population $\vec{\boldsymbol{X}}(t)$ are optimal. As a result, the population $\vec{\boldsymbol{X}}(t)$ would still contain some other optimal individual $\hat{\boldsymbol{X}}^*$ even if the optimal individual $\boldsymbol{X}^*$ is removed by the last struggle operator. So the population $\vec{\boldsymbol{X}}(t)$ is still an optimal population. Therefore, for the population Markov chain $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ generated by LDSE, the optimal state set Γ^* is a closed set.

(2) Note that all of the reproduction operators in basic LDSE, including reflection, contraction and last struggle are linear. Once all individuals in the population are degenerated to a subspace of $\mathbb{R}^n$, then no offspring individual can get out of this subspace, even though the evolutionary process continues infinitely. Therefore, the state set corresponding to some populations distributed in the same subspace of $\mathbb{R}^n$ is a closed set.

(3) Because all the reproduction operators in basic LDSE are linear, so all individuals in a uniform population would remain unchanged in the evolutionary process of basic LDSE. As a result, the population can never get out of the uniform population. That is, the state set corresponding to uniform populations is a closed set. ■

The above result, optimal state set Γ^* is a closed set, indicates that LDSE can preserve elite information, although an elite individual might be replaced by another elite one. Theorem 4.4(2) and (3) indicate that basic LDSE is not guaranteed to converge to the global minimum. Despite this, basic LDSE is still applicable. In fact, basic LDSE has a good balance between the global

and local search, and numerical results show that basic LDSE converge to the global optimum very fast in many applications.

Next, we will show that modified LDSE is globally convergent. Note that the NS operator enables modified LDSE to search any point in the search space $\mathbf{\Omega}$. As a result, LDSE has an ergodic property in some sense. In fact, normal struggle operator enables the new individual to escape from the subspace spanned by all individuals in the current population, even if the current population is uniform. Theorem 4.4(1) indicates that the ergodic property of LDSE is not complete since the optimal state set Γ^* is a closed set. In the state space $\Gamma = \{S_1, S_2, \ldots, S_M\}$, without loss of generality, it can be assumed that the first k states are optimal state and the remains are non-optimal, i.e.

$$S_i \in \begin{cases} \Gamma^*, & i = 1, 2, \ldots, k; \\ \Gamma \backslash \Gamma^*, & i = k+1, k+2, \ldots, M. \end{cases}$$

Let p_{ij} be the probability of going from state S_i to state S_j and the matrix $P = (p_{ij})_{M\times M}$ be the transition matrix of the Markov chain generated by modified LDSE. Based on the above analysis, the transition matrix $P = (p_{ij})_{M\times M}$ has the following form:

$$P = \begin{pmatrix} p_{1,1} & \cdots & p_{1,k} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{k,1} & \cdots & p_{k,k} & 0 & \cdots & 0 \\ p_{k+1,1} & \cdots & p_{k+1,k} & p_{k+1,k+1} & \cdots & p_{k+1,M} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{M,1} & \cdots & p_{M,k} & p_{M,k+1} & \cdots & p_{M,M} \end{pmatrix},$$

where k is the number of optimal state, $0 < p_{ij} < 1$, for $i = 1, 2, \ldots, k$, $j = 1, 2, \ldots, k$, and $\sum_{j=1}^{k} p_{ij} = 1, i = 1, 2, \ldots, k$. $0 < p_{ij} < 1$, for $i = k+1, \ldots, M, j = 1, 2, \ldots, M$, and $\sum_{j=1}^{M} p_{ij} = 1, i = k+1, k+2, \ldots, M$.

The above transition matrix can be partitioned into four blocks, i.e, $P = \left(\begin{smallmatrix} P_1 & O \\ P_2 & P_3 \end{smallmatrix}\right)$, where P_1 is a $k \times k$ matrix, P_2 is an $(M-k) \times k$ matrix, O is a $k \times (M-k)$ zero matrix, P_3 is an $(M-k) \times (M-k)$ matrix.

Before the proof of convergence, we give some relevant conclusions.

LEMMA 4.5 (Perron Frobenius theorem [15]) *Let $A = (a_{ij})_{n\times n}, a_{ij} \geq 0$ be irreducible, then the following propositions are true.*

(1) $\lambda = \rho(A)$ *and* $\lambda > 0$;
(2) *the algebraic multiple of* λ *is* 1.

where λ, $\rho(A)$ *is the eigenvalue, spectral radius of matrix A, respectively.*

LEMMA 4.6 (Gerschgorin circular disc theorem [15]) *Let λ be an eigenvalue of an arbitrary matrix $A = (a_{ij})_{n\times n} \in \mathbb{C}^{n\times n}$, then $\lambda \in \bigcup_{i=1}^{n} D_i$, where $D_i = \{z \in \mathbb{C} \mid |z - a_{ii}| \leq r_i\}, r_i = \sum_{j\neq i} |a_{ij}|$.*

LEMMA 4.7 *Let A be a stochastic matrix, i.e., $A = (a_{ij})_{n\times n} \in \mathbb{R}^{n\times n}, 0 \leq a_{ij} \leq 1, \sum_j a_{ij} = 1$, then A to the power of t, A^t, is also a stochastic matrix for any positive integer t.*

Proof First, we will prove that $C = A^2 = A \cdot A$ is a stochastic matrix.

In fact, it is obvious that $0 \leq c_{ij} \leq 1$, where $c_{ij} = \sum_{k=1}^{n} a_{ik}a_{kj}$, and the sum of the ith row of C satisfies $c_{i\cdot} = \sum_j^n c_{ij} = \sum_j^n \sum_{k=1}^n a_{ik}a_{kj} = \sum_k^n \sum_{j=1}^n a_{ik}a_{kj} = \sum_k^n (a_{ik} \cdot \sum_{j=1}^n a_{kj}) = \sum_k^n a_{ik} = 1$. Therefore, C is a stochastic matrix.

Based on the fact that A^2 is a stochastic matrix, it is easy to show that $B = A^t$ is still a stochastic matrix by mathematical induction. ■

Next, we will study the limit of P^t as $t \to \infty$.

THEOREM 4.8 *For the above transition matrix P, the following statements are true:*

(1) P_1 *has only one simple eigenvalue* 1, *and the modulus of the other eigenvalues are all less than* 1.
(2) *The modulus of all eigenvalues of* P_3 *are less than* 1.

Proof (1) Note that the matrix P_1 satisfies $\sum_{j=1}^{k} p_{ij} = 1$ $(i = 1, 2, \ldots, k)$, and $0 < p_{ij} < 1$ $(i = 1, 2, \ldots, k; j = 1, 2, \ldots, k)$, so P_1 is nonnegative and irreducible.

It is obvious that $P_1 \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}$, so 1 is an eigenvalue of P_1. Thus, $\rho(P_1) \geq 1$.

On the other hand, since $\rho(P_1) \leq ||P_1||_\infty = 1$, we have $\rho(P_1) = 1$. By Perron–Frobenius theorem, 1 is a simple eigenvalue of P_1.

Note that Gerschgorin discs D_i, $i = 1, 2, \ldots, k$ are all tangent to the unit circle in the complex plane, and they have a unique tangent point $1 + 0 \cdot i$ (see Figure 6). By Gerschgorin circular disc theorem, the modulus of all eigenvalues of P_1 are less than or equal to 1. Since 1 is its simple eigenvalue, the modulus of the other eigenvalues are all less than 1.

(2) The matrix P_3 satisfies $\sum_{j=1}^{k} p_{ij} < 1$, $(i = k+1, k+2, \ldots, M)$, and $0 < p_{ij} < 1, (i = k+1, k+2, \ldots, M, j = k+1, k+2, \ldots, M)$. So the spectral radius of the matrix $\rho(P_3) < 1$. Thus, the modulus of all eigenvalues of P_3 are less than 1. ■

THEOREM 4.9 *Let* P^t *be the t-power of the above matrix P, then the limit of* P^t *exists as* $t \to \infty$, *and the limit matrix* P^∞ *is a stable matrix of the form*

$$P^\infty = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} (p_1, p_2, \ldots, p_k, 0, 0, \ldots, 0), \quad \sum_{i=1}^{k} p_i = 1.$$

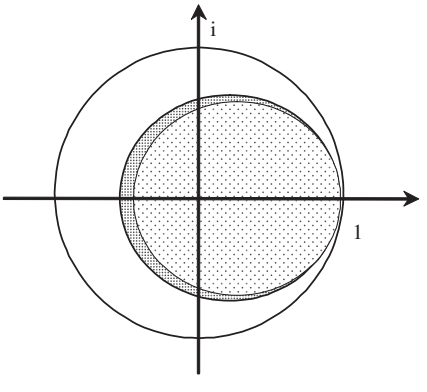

Figure 6. Gerschgorin discs of Matrix P_1, only two discs are shown.

Proof The Jordan normal form of P can be described as

$$P = H^{-1}\begin{pmatrix} \lambda_1 & \sigma_1 & & \\ & \lambda_2 & \ddots & \\ & & \ddots & \sigma_{M-1} \\ & & & \lambda_M \end{pmatrix} H,$$

where $\sigma_i \in \{0, 1\}$. It can be assumed that $|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_M|$. By Theorem 4.8(1), P_1 has only one simple eigenvalue 1, and the modulus of the other eigenvalues are all less than 1. By Theorem 4.8(2), the modulus of all eigenvalues of P_3 are all less than 1. So P can be rewritten as

$$P = H^{-1}\begin{pmatrix} 1 & \sigma_1 & & \\ & \lambda_2 & \ddots & \\ & & \ddots & \sigma_{M-1} \\ & & & \lambda_M \end{pmatrix} H,$$

where $|\lambda_i| < 1$, $i = 2, \ldots, M$. Therefore,

$$P^t = H^{-1}\begin{pmatrix} 1 & \sigma_1 & & \\ & \lambda_2 & \ddots & \\ & & \ddots & \sigma_{M-1} \\ & & & \lambda_M \end{pmatrix}^t H.$$

Thus the limit of P^t exists. Let $\lim_{t\to\infty} P^t = P^\infty$, then

$$P^\infty = H^{-1}\begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{pmatrix} H.$$

Let

$$H = \begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ h_{21} & h_{22} & \cdots & h_{2M} \\ \cdots & \cdots & \cdots & \cdots \\ h_{M1} & h_{M2} & \cdots & h_{MM} \end{pmatrix}, \quad H^{-1} = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1M} \\ g_{21} & g_{22} & \cdots & g_{2M} \\ \cdots & \cdots & \cdots & \cdots \\ g_{M1} & g_{M2} & \cdots & g_{MM} \end{pmatrix},$$

then

$$P^\infty = H^{-1}\begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{pmatrix} H = H^{-1}\begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{pmatrix}\begin{pmatrix} 1 & & & \\ & 0 & & \\ & & \ddots & \\ & & & 0 \end{pmatrix} H$$

$$= \begin{pmatrix} g_{11} & 0 & \cdots & 0 \\ g_{21} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ g_{M1} & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} = \begin{pmatrix} g_{11} \\ g_{21} \\ \vdots \\ g_{M1} \end{pmatrix}(h_{11} \quad h_{12} \quad \cdots \quad h_{1M}).$$

Meanwhile, note that

$$P^t = \begin{pmatrix} P_1 & O \\ P_2 & P_3 \end{pmatrix}^t = \begin{pmatrix} P_1^t & O \\ X & P_3^t \end{pmatrix},$$

then

$$P^\infty = \begin{pmatrix} \hat{A} & O_{k\times(M-k)} \\ \hat{X} & O_{(M-k)\times(M-k)} \end{pmatrix}.$$

The result indicates that the columns after the kth column in P^∞ are all zeros. Therefore,

$$P^\infty = \begin{pmatrix} g_{11} \\ g_{21} \\ \vdots \\ g_{M1} \end{pmatrix} (h_{11} \quad \cdots \quad h_{1k} \quad 0 \quad \cdots \quad 0) = \begin{pmatrix} g_{11}h_{11} & \cdots & g_{11}h_{1k} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ g_{M1}h_{11} & \cdots & g_{M1}h_{1k} & 0 & \cdots & 0 \end{pmatrix}.$$

It is easy to show that P^∞ is still a stochastic matrix by Lemma 4.7. Consequently,

$$g_{i1}h_{11} + \cdots + g_{i1}h_{1k} = g_{i1} \cdot (h_{11} + \cdots + h_{1k}) = 1, \quad i = 1, 2, \ldots, M.$$

So

$$g_{11} = g_{21} = \cdots = g_{M1}.$$

Define

$$g_{11}h_{11} = \cdots = g_{M1}h_{11} = p_1,$$

$$\vdots$$

$$g_{11}h_{1k} = \cdots = g_{M1}h_{1k} = p_k.$$

Then

$$P^\infty = \begin{pmatrix} p_1 & \cdots & p_k & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ p_1 & \cdots & p_k & 0 & \cdots & 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} (p_1, p_2, \ldots, p_k, 0, 0, \ldots, 0)$$

and

$$\sum_{i=1}^{k} p_i = 1. \qquad \blacksquare$$

THEOREM 4.10 *For any initial population $\vec{X}(0)$, modified LDSE converges to the global optimum.*

Proof Let $\boldsymbol{q}^{(t)}$ be the state distribution of population $\vec{\boldsymbol{X}}(t)$, P be the transition matrix of population sequence generated by modified LDSE, then we have

$$\begin{aligned}
\boldsymbol{q}^{(1)} &= \boldsymbol{q}^{(0)} \cdot P, \\
\boldsymbol{q}^{(2)} &= \boldsymbol{q}^{(1)} \cdot P = \boldsymbol{q}^{(0)} \cdot P^2, \\
&\vdots \\
\boldsymbol{q}^{(t)} &= \boldsymbol{q}^{(t-1)} \cdot P = \boldsymbol{q}^{(0)} \cdot P^t, \\
&\vdots
\end{aligned}$$

By Theorem 4.9, the limit matrix P^∞ exists, and it has the following form:

$$P^\infty = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} (p_1, p_2, \ldots, p_k, 0, 0, \ldots, 0)$$

and

$$\sum_{i=1}^{k} p_i = 1.$$

So, we have

$$\lim_{t\to\infty} \boldsymbol{q}^{(t)} = \boldsymbol{q}^{(0)} \cdot P^\infty = \boldsymbol{q}^{(0)} \cdot \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} (p_1, p_2, \ldots, p_k, 0, 0, \ldots, 0).$$

Note that $\boldsymbol{q}^{(0)}$ is a distribution vector, so $0 \le q_i \le 1$, and $\sum_{i=1}^{M} q_i = 1$. Thus,

$$\lim_{t\to\infty} \boldsymbol{q}^{(t)} = (p_1, p_2, \ldots, p_k, 0, 0, \ldots, 0).$$

Therefore, for any initial distribution $\boldsymbol{q}^{(0)} = (q_1, q_2, \ldots, q_M)$, we have

$$\boldsymbol{q}^\infty = (p_1, p_2, \ldots, p_k, 0, 0, \ldots, 0).$$

Note that the first k states $S_i, i = 1, 2, \ldots, k$ belongs to optimal state set Γ^* and the latest $M-k$ states $S_i, i = k+1, k+2, \ldots, M$ are non-optimal. That is, $\boldsymbol{q}^\infty$ satisfies

$$q_i^\infty = \begin{cases} 0 & \text{if } S_i \notin \Gamma^* \\ p_i \ge 0 & \text{if } S_i \in \Gamma^* \end{cases} \quad \text{for all } i = 1, 2, \ldots, M$$

and

$$\sum_{S_i \in \Gamma^*} q_i^\infty = 1.$$

By Lemma 4.2 on p. 20, modified LDSE converges to the global optimum for any initial population $\vec{\boldsymbol{X}}(0)$. ■

4.2 *Estimate of convergence speed*

In this subsection, we give an estimate of the convergence speed of modified LDSE by Markov stopping time.

DEFINITION 4.11 *Let* $\{\vec{\boldsymbol{X}}(t)\}_{t=0}^{\infty}$ *be the population Markov chain generated by modified LDSE, then*

$$T = \min\{t \in \mathbb{N} | \vec{\boldsymbol{X}}(t) \in \Gamma^*\}$$

is called the first passage time of the Markov chain.

THEOREM 4.12 *Let* T *be the first passage time of the population Markov chain generated by modified LDSE, then its mathematical expectation* $E(T)$ *satisfies* $E(T) \leq T_0$, *where* $T_0 = \max\{t_0 + 2, (M-k)/(1-p_*) \cdot \sum_{j=1}^{k} p_{i_0 j}\}, t_0$, p_* *are constants,* $t_0 \in \mathbb{N}$, $0 < p_* < 1$, $i_0 \in \{k+1, k+2, \ldots, M\}$.

Proof To estimate the mathematical expectation of the first passage time $E(T)$, we reduce the transition matrix. Similar to the above section, the state space is divided into two parts: $\Gamma = \Gamma_1 \cup \Gamma_2$, where $\Gamma_1 = \Gamma^*, \Gamma_2 = \Gamma \backslash$ï¸Ł$\Gamma^*$. Let d_{ij} be the probability of transition from state set Γ_i to state set Γ_j, then $D = (d_{ij})_{2\times 2}$ is the transition matrix between the state sets Γ_i and Γ_j of the population Markov sequence. By Theorem 4.4, the state in Γ_1 will never transit to the state in Γ_2, so $d_{11} = 1, d_{12} = 0$. Let $d_{21} = x$, then $d_{22} = 1 - x$.

Let T_{ij} be the first passage time of the population Markov sequence of the state sets. It is obvious that $E(T) \leq E(T_{21})$.

Since $P\{T_{21} = t\} = x(1-x)^{t-1}$, we have

$$E(T_{21}) = \sum_{t=1}^{\infty} t \cdot x \cdot (1-x)^{t-1} = x \cdot (-1)\frac{\mathrm{d}}{\mathrm{d}x}\left(\sum_{t=1}^{\infty}(1-x)^t\right) = -x \cdot \frac{\mathrm{d}}{\mathrm{d}x}\left(\frac{1-x}{x}\right) = \frac{1}{x}.$$

Thus

$$E(T) \leq \frac{1}{x}.$$

Let $p_i^{t_0}$ be the probability that the population is in state S_i at the time t_0, $i = 1, 2, \ldots, M$, then

$$\sum_{i=1}^{M} p_i^{t_0} = 1.$$

Let $p_*^{t_0} = \sum_{i=1}^{k} p_i^{t_0}$, then $p_*^{t_0}$ is the probability that the population is in an optimal state.

(1) If there exists a $p_* \in (0,1)$, such that $p_*^{t_0} \leq p_* < 1$, then

$$\sum_{i=k+1}^{M} p_i^{t_0} \geq 1 - p_*.$$

It is obvious that there exists an $i_0 \in \{k+1, k+2, \ldots, M\}$, such that $p_{i_0}^{t_0} > (1-p_*)/(M-k) > 0$.

By the formula of total probability, we have

$$x = \sum_{i=k+1}^{M}\left(p_i^{t_0} \cdot \sum_{j=1}^{k} p_{ij}\right) \geq p_{i_0}^{t_0} \cdot \sum_{j=1}^{k} p_{i_0 j} \geq \frac{1-p_*}{M-k}\sum_{j=1}^{k} p_{i_0 j}.$$

Therefore, $E(T) \leq (M-k)/(1-p_*) \cdot \sum_{j=1}^{k} p_{i_0 j}$.

(2) If the above p_* does not exist, then there exists a r, $0.5 < r < 1$, such that $p_*^{t_0} \geq r$.

Since p_*^t is monotone nondecreasing with the generation t, if $\tau > t_0$, then $p_*^\tau \geq r$.
Therefore, $P\{T = \tau\} \leq r(1-r)^{\tau-1}$.
Thus, $E(T) \leq t_0 + \sum_{\tau=1}^{\infty} r \cdot \tau \cdot (1-r)^{\tau-1} = t_0 + 1/r \leq t_0 + 2$.
As a result, we have $E(T) \leq \max\{t_0 + 2, (M-k)/(1-p_*) \cdot \sum_{j=1}^{k} p_{i_0 j}\}$. ■

The above theorem gives an upper bound of the mathematical expectation $E(T)$ of the first passage time T. By Markov's inequality, we have

$$P\{T > \tilde{T}\} \leq \frac{E(T)}{\tilde{T}}.$$

If $E(T) \leq \tilde{T}/\mu$, then

$$P\{T > \tilde{T}\} \leq \frac{1}{\mu}.$$

The result can be explained as follows. If modified LDSE is executed λ times independently, then the probability that the population can reach the optimal state before the $\tilde{T}$th step at least one time is not less than $\tilde{p} = 1 - (1/\mu)^\lambda$. In the case of $\mu = 4$, $\lambda = 10$, $\tilde{p} = 1 - 9.53674 \times 10^{-7} < 10^{-6}$. This means that if we set $\tilde{T} = 4 \cdot T_0$ and modified LDSE are applied 10 times, then the probability that the population can reach the optimal state before the $\tilde{T}$th step at least one time is not less than $1 - 10^{-6}$.

5. Numerical results

We have introduced three techniques and four modified versions of LDSE. It can be seen from Section 3 that their relations are:

- LDR + basic LDSE = LDR-LDSE;
- NS + LDR + basic LDSE = NS-LDSE;
- VD + LDR + basic LDSE = VD-LDSE;
- LDR + NS + VD + basic LDSE = modified LDSE;
- VD-LDSE = LDR-LDSE if in case $m = 2$;
- LDR-LDSE = basic LDSE if we set $p_a = 0.0$.

In this section, we will (1) show you the performance of these algorithms; (2) study the effect of each technique; and (3) discuss the choice of control parameters for modified LDSE.

5.1 *Comparison of LDSEs and DERL*

To compare with other evolutionary algorithms, we choose DE algorithm with random localization (DERL) [10], which is an improved version of DE. We choose it because (1) DE is a widely used GO algorithm, and DERL is reported to have improved the performance of DE significantly; (2) LDSEs and DERL share similar method for parent selection, and both of them make use of function–value information of selected parents while generating offsprings. In fact, Both algorithms reproduce new individuals from the ith individual and some randomly selected parents. LDSEs need to find the best and the worst parents to apply the simplex operations and DERL also

needs to find the best parent to conduct the mutation operation. In DE/bin/1, the mutated point is generated in the following way:

$$X_{\mathrm{m}}(t) = X_{r_1}(t) + F \cdot (X_{r_2}(t) - X_{r_3}(t))$$

DERL requires that the base point $X_{r_1}(t)$ is the best and the scaling factor F is uniformly drawn from $[-1, -0.4] \cup [0.4, 1]$. The source code of DERL used in this section is adapted from the C code (de36.C) downloaded from the home page of R. Storn. The adaption follows paper [10].

Two groups of test functions are used to assess the performance of concerned algorithms. The first group consists of 10 dimension-adjustable (that is, n is an adjustable parameter of the functions) problems with box-constraints selected from the test suite in [1], which are frequently used for testing the performance of evolutionary algorithms. Only dimension-adjustable problems are selected because we want to use them to study the influence of problem dimension n on modified LDSE. The second group consists of 12 benchmark functions from the CEC'2005 Special Session on real-parameter optimization [22]. There are 25 problems altogether in the CEC'2005 test suite. The selected benchmark functions are their first 12. The last 13 problems are omitted in this paper because all our concerned algorithms (including modified LDSE and DERL) failed to find global optimum for them.

The stopping criteria are set as follows. The algorithm will stop if any one of the following three conditions is satisfied: (1) the global minimum is attained, i.e. $f_{GloBest}(t) - f^* < \varepsilon$, where $\varepsilon = 10^{-6}$, and f^* is its known global minimum; (2) the population is matured, i.e. $f_{PopWorst}(t) - f_{\mathrm{PopBest}}(t) < \delta$, where $\delta = 10^{-4}$; (3) the maximum number of function evaluations E is reached, where E is set to $E = n^2 \times 10^4$ in our numerical experiments. A run is said to be successful if and only if it satisfies condition (1). Otherwise, we say the algorithm is failed in this run. The percentage of success (ps) and the averaged number of function evaluations (nfe) required to find the global minimum are recorded to show the performance (reliability and efficiency) of each algorithm. To reduce the influence of randomness, 100 replications are done for each problem and each algorithm with different random seeds, and the results presented in this section are all averaged values. The statistical processes are done within the programs automatically.

As known to all, the choice of control parameters in an algorithm may affect its efficiency and reliability. Finding appropriate parameter values for the algorithm is sometimes a key step in practical implementations. The more the parameters in the algorithm, the harder to tune. So, we restrict the number of parameters as few as possible for modifications of LDSE to make them easy to implement. Some parameters are treated as implicit parameters. They will be tuned to fixed values in advance. For example, in our numerical experiments, the scale factor α and β for modified LDSE are fixed to $\alpha = 1$, $\beta = \frac{1}{3}$; the mean and the variance of normal distribution $\mathrm{N}(\mu, \sigma^2)$ are fixed to $\mu = 0$ and $\sigma = (u - l)/3$. The others (explicit parameters) are left for users to tune. In this way, only one control parameter, the adsorption probability p_{a} for using LDR technique, is added. Therefore, modified LDSE has only three explicit parameters to tune: population size N, adsorption probability p_{a}, and simplex dimension m. It means that no additional control parameter is needed for using NS and/or VD techniques.

In our experiments, the population size N is tuned empirically in a simple way. First we do a series of numerical experiments with different parameter values (e.g. with step 100) in a wide range (10–2000) to find an appropriate range. This is a coarse tuning process. Then we do another series of numerical experiments with different parameter values (e.g. with step 10) within the range attained by the coarse turning. This is a fine tuning process. Other parameters are tuned in a similar way, but they are fixed to some discrete values empirically. For example, adsorption probability p_{a} is confined to $\{0.1, 0.2, 0.8, 0.9\}$ and simplex dimension m is confined to $\{2, 3, 4\}$. Note that the scaling factor F is a preset parameter in the original DE, but it becomes a random number chosen from the interval $[-1, -0.4] \cup [0.4, 1]$ in DERL. So DERL has only two parameters: the

Table 1. Comparison of LDSEs and DERL using the first group of test problems.

Problem		Basic LDSE						LDR-LDSE	NS-LDSE	VD-LDSE	Modified LDSE	DERL		
No	Name	N	m	nfe	ps	N	p_a	nfe	nfe	nfe	nfe	N	P_c	nfe
1	ACK	800	4	39,809	74	40	0.8	35,593	33,974	32,043	**28,795**	40	0.1	40,271
2	CM	400	3	19,445	100	40	0.2	12,113	9286	11,608	**8748**	40	0.5	13,689
3	GW	800	3	15,007	100	30	0.8	13,957	12,396	10,877	**10,053**	40	0.1	22,391
4	LM1	1000	3	8487	100	40	0.2	**7486**	7976	7511	7643	40	0.5	12,964
5	LM2	800	3	10,557	100	30	0.8	7283	61,154	64,856	**6109**	40	0.5	12,687
6	NF3	400	2	559,892	100	100	0.1	285,380	337,863	285,380	**276,872**	100	0.1	874,017
7	RB	600	3	1,179,414	25	80	0.1	620,677	367,079	522,714	317,951	80	0.9	**174,193**
8	RG	600	3	33,561	100	50	0.8	27,983	27,553	27,012	**26,935**	50	0.1	38,664
9	SWF	800	2	58,485	100	50	0.8	40,908	31,594	40,908	**21,830**	50	0.1	23,098
10	SIN	100	3	17,530	100	50	0.1	14,012	13,088	13,765	**12,769**	50	0.5	18,456

Problem dimension $n = 20$. Control parameters and performance indicators, including percentage of success (ps), the averaged number of function evaluations (*nfe*) required to reach the global optimum with the given precision ($f^* + 10^{-6}$) are listed. All modified versions of LDSE share the same control parameter m. NS-LDSE, VD-LDSE, and modified LDSE share the same control parameters N and p_a as LDR-LDSE. The least required nfe are marked in bold.

Table 2. Number of points of each algorithm on each problem according to the result of Table 1.

No	Basic LDSE	LDR-LDSE	NS-LDSE	VD-LDSE	Modified LDSE	DERL
1	2	3	4	5	6	1
2	1	3	5	4	6	2
3	2	3	4	5	6	1
4	2	6	3	5	4	1
5	2	3	5	4	6	1
6	2	4	3	5	6	1
7	1	2	4	3	5	6
8	2	3	4	5	6	1
9	1	2	4	3	6	5
10	2	3	5	4	6	1
Total	17	32	41	43	57	20

Scoring method: Start with 1 point, add 1 point if an algorithm wins another one in terms of averaged *nfe* for the same problem.

population size N and the crossover probability P_c. The population size N is tuned in the same way as that of LDSEs. And the crossover probability P_c is confined to $\{0.1, 0.5, 0.9\}$. Only the best parameter combination for each algorithm to each test problem is presented in Tables 1 and 3. These parameter values might not be optimal, but they should be good ones.

In Table 1, percentage of success (ps) is omitted except for basic LDSE because all modified versions of LDSE and DERL have solved the problems with ps = 100%. From the sixth column of Table 1, we can see that basic LDSE might fail to solve problem ACK and RB with the listed population size N. One can get a better ps by giving a larger population size, but the success performance (*nfe*/*ps*) will decrease dramatically. From the third and seventh columns of Table 1, we can see that the population size N of basic LDSE is usually much larger than that of modified versions of LDSE for the same problem. In this test, to get a good success performance, N is selected from [100, 1000] for basic LDSE, and [30, 100] for modified versions of LDSE. The result confirms that basic LDSE needs a large population to ensure its convergence because their reproduction operators are linear, and LDR technique does work to reduce the population size. We believe that this is the main reason why LDR-LDSE (and its followers) converge faster than basic LDSE.

Numerical experiments indicate that modified versions of LDSE can share similar population size N, adsorption probability p_a, and simplex dimension m, for a given problem. The sharing does not have much influence on their performance. So all modified versions of LDSE use the same control parameters in our test. The parameters are tuned by LDR-LDSE and shared by the others.

Non-parametric tests can be used here to compare the performance of involved algorithms. For example, S. Garcia et al recommend using Wilcoxon signed-rank test for detecting the differences between two algorithms, and Holm and Hochberg procedures for multiple comparisons [6]. Non-parametric tests are scientifically informative, but they are difficult to understand for those who are not familiar with statistics. In this work, we define a scoring method (see Table 2) and count the number of points according to the result of Table 1. The performance of an algorithm is simplified as its number of points scored from competitions. It is easier to understand. The simplified performance is listed in Table 2.

We can see that basic LDSE gets the least number of points 17. It performs worst. LDR-LDSE, NS-LDSE and VD-LDSE get 32, 41 and 43 points, respectively. NS-LDSE performs better than LDR-LDSE. This result confirms that NS technique does work to enhance the local search ability of LDR-LDSE. VD performs better than LDR-LDSE. This result confirms that drawing lessons from the recent failure by searching on the sub-facet does help to accelerate its convergence speed. All modified versions of LDSE perform better than basic LDSE. Modified LDSE gets the most

Table 3. Comparison of modified LDSE and improved DE using CEC'2005 test functions, problem dimension $n = 10$.

	DERL							Modified LDSE							
No	N	P_c	*nfe*_min	*nfe*_max	*nfe*_ave	*nfe*_std	ps	N	m	p_a	*nfe*_min	*nfe*_max	*nfe*_ave	*nfe*_std	ps
1	50	0.1	11,773	13,188	12,482	332.7	100%	50	2	0.1	4084	6505	5573	361.74	100%
2	50	0.9	19,295	21,679	20,427	873.64	100%	50	2	0.1	10,019	12,941	11,218	891.87	100%
3	50	0.9	35,930	42,508	39,073	2263.55	100%	50	2	0.1	232,158	349,915	297,509	36949.21	100%
4	50	0.5	67,243	83,209	75,379	3198.61	100%	50	2	0.1	10,322	61,216	15,392	5154.9	100%
5	50	0.9	50,333	55,696	52,638	1645.02	100%	50	2	0.1	26,171	30,451	27,481	1197.27	100%
6	50	0.9	26,982	132,655	36,385	22301.74	95%	50	2	0.1	33,791	139,450	47,169	32481.53	98%
7	/300	/0.9			Failed		0	300	3	0.1	33,127	42,589	36,619	4133.32	47%
8	/300	/0.9			Failed		0	/300	/3	/0.9			Failed		0
9	50	0.5	16,508	18,085	17,584	2546.05	100%	50	3	0.1	19,189	29,841	27,534	7869.58	86%
10	/300	/0.9			Failed		0	200	4	0.1	37,663	13,652	54,247	8359.22	23%
11	50	0.9	76,087	90,551	82591	17341.11	100%	200	2	0.1	114,346	121,651	117,998	5165.41	74%
12	50	0.9	23,481	229,142	50,728	12097.17	76%	200	2	0.1	27,605	11,358	53,433	1073.24	36%

Control parameters and performance indicators, including percentage of success (ps), the minimum, averaged and maximum number of function evaluations (*nfe*_min, *nfe*_ave, *nfe*_max) and its standard deviation (*nfe*_std) are listed, where $N = /300$, $P_c = /0.9$ means all possible combinations of $N = 50/100/200/300$ and $P_c = 0.1/0.5/0.9$ are tested for DERL, and $N = /300$, $p_a = /0.9$, $m = /4$ means all possible combinations of $N = 50/100/200/300$, $P_a = 0.1/0.9$ and $m = 2/3/4$ are tested for modified LDSE.

number of points 57. It has the best performance for this group of test problems. This result shows that LDR, NS and VD techniques can work well together to improve the performance of LDSE. The improved DERL gets 20 points. It is only a little better than basic LDSE. Modified LDSE outperforms DERL considerably on this group of test problems.

To make a further comparison, we apply modified LDSE and DERL to the second group of test problems. The test result is shown in Table 3. We can see that DERL is failed to solve 3 problems (including the 7th, 8th and 10th problem) within the limited number of *nfe*. Modified LDSE also failed to solve the 8th problem, but it can solve all other problems with positive ps. For the 9th, 11th and 12th problem, DERL performs better for its higher ps. For the first five problems, modified LDSE performs better than DERL for its less *nfe*.

5.2 *Choice of control parameters for modified LDSE*

As previously described, the choice of control parameters in an algorithm may affect its efficiency and reliability. In this section, we will demonstrate the effect of the three explicit parameters in modified LDSE: population size N, adsorption probability p_a, and simplex dimension m, respectively. This might be helpful to choose appropriate parameter values.

The experience of numerical experiments shows that increasing population size N can definitely improve the percentage of success (ps) if the maximum number of function evaluations E is sufficient large. The required number of function evaluations (*nfe*) will increase almost linearly with the increase of population size N (see Figure 7(b)). Decreasing *nfe* can accelerate the convergence speed at the risk of getting stuck in a non-optimal point. Usually, $N = 2 \cdot n$ is a first choice. If it works well, you can reduce it in the hope of a faster convergence. A reasonable choice might be $1.5 \cdot n$. However, be aware that N should be greater than the problem dimension n in general. If $N = 2 \cdot n$ does not work, try to increase N for a larger percentage of success. The reasonable choice might be $5 \cdot n$, $10 \cdot n$, $20 \cdot n$, etc. Usually, N should not be larger than $30 \cdot n$.

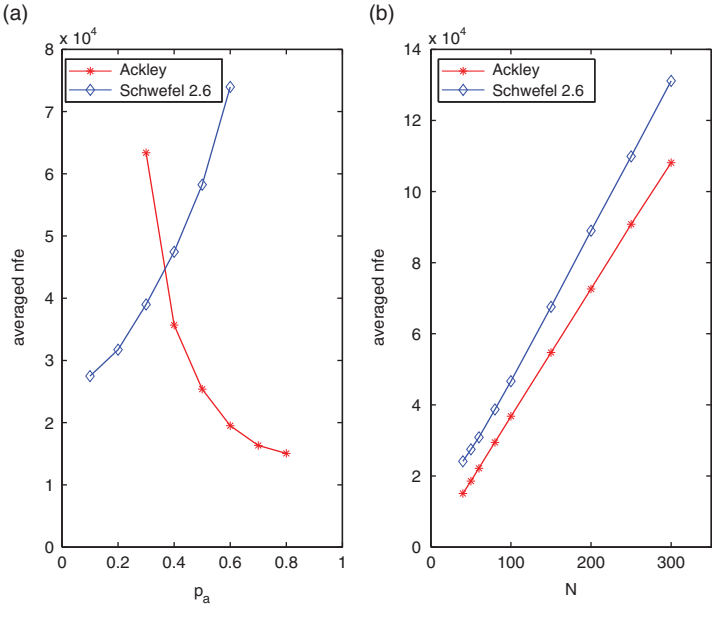

Figure 7. Effect of different parameters. Problem dimension $n = 10$. Population size N is fixed ($N = 40$) in case of (a), and adsorption probability p_a is fixed ($p_a = 0.8$ for ACK and $p_a = 0.1$ for Schwefel 2.6) in case of (b).

Table 4. Choice of simplex dimension m.

Problem		Problem dimension $n = 10$					Problem dimension $n = 20$					Problem dimension $n = 50$				
No	Name	N	p_a	$m = 2$	3	4	N	p_a	$m = 2$	3	4	N	p_a	$m = 2$	3	4
1	ACK	30	0.8	15,112	15,838	11,189	40	0.9	32,533	35,261	**28,795**	80	0.9	81,013	**72,716**	79,646
2	CM	30	0.8	**3322**	3450	5641	40	0.9	10,644	**8748**	90,781	80	0.8	20,639	**17,334**	22,665
3	GW	30	0.8	**7635**	8169	86301	30	0.8	11,228	**10,053**	12,566	80	0.8	31,614	29,458	**26,296**
4	LM1	20	0.8	**2708**	4383	5370	40	0.8	7762	**7543**	8019	80	0.8	25,362	23,106	**18,784**
5	LM2	30	0.8	**4394**	5262	5631	30	0.8	6864	**6309**	7297	80	0.8	**28,173**	31,127	30,481
6	NF3	80	0.1	**15,766**	18,867	21,221	100	0.1	316,971	**276,872**	423,775	100	0.1	1,672,537	1,417,112	**1,176,943**
7	GB	50	0.1	125,515	137,833	89,905	80	0.1	670,356	**317,951**	583,917	80	0.1	1,281,485	**893,804**	1,431,299
8	RG	30	0.8	17746	**9056**	9308	50	0.8	31,442	**26,935**	27,405	80	0.8	130,710	96,239	**704,284**
9	SWF	30	0.8	25,852	**12,587**	13,609	50	0.8	21,830	28,162	32,216	80	0.8	**275,351**	316,574	395,343
10	SIN	20	0.8	**2546**	4951	7361	50	0.8	12,927	**12,769**	13,398	100	0.8	392,356	**385,031**	429,787
Tol. points				**25**	18	17			18	**27**	15			17	**23**	20

Control parameters and averaged number of function evaluations (*nfe*) required to reach the global optimum with the given precision ($f^* + 10^{-6}$) using different control parameter m for the problems with different numbers of dimension n are listed. The total points (in the last line) are scored using the same scoring method as that of Table 2. The least required nfe are marked in bold.

The choice of adsorption probability p_a is problem dependent (see Figure 7(a)). It ranges from 0.1 to 0.95 in our experiments. But in most cases, it is a good choice to set $p_a = 0.1/0.2$, where $p_a = 0.1/0.2$ denotes $p_a = 0.1$ or $p_a = 0.2$. Similar notations are used anywhere in this paper. For example, it works well for all test functions from the second test group (see Table 3) and a half of the first test group (see Table 1). In some cases, $p_a = 0.8/0.9$ works well, e.g. for problem ACK, GW, LM2, RG and SWF. Usually, $p_a = 0.5$ is a safe choice. If you care about ps more than *nfe*, just choose it. The best adsorption probability p_a is not sensitive to problem dimension n for a given function structure. For example, if $p_a = 0.8$ works well for ACK in case $n = 10$, you can expect a good result with $p_a = 0.8/0.9$ for ACK in case $n = 20/50$ (see Table 4). It is likely to get a bad result with $p_a = 0.1/0.2$.

For a given problem, if $p_a = p_0$ works the best, reducing/increasing will increase the required *nfe* nonlinearly (see Figure 7(a)).

Simplex dimension m is another control parameter in modified LDSE. Numerical experiments indicate that the optimal simplex dimension m^* depends on problem dimension n. To find it out, we conduct a series of experiments on problems with different numbers of dimension n using different control parameter m. Control parameter N, p_a, and maximum *nfe* are set such that all algorithm can find the global optimum with $ps = 100\%$. The averaged *nfe* required to reach the global optimum with the given precision ($f^* + 10^{-6}$) is recorded to show the convergence speed of a given combination (n and m). The test result is listed in Table 4. Number of points is also counted to simplify the comparison. We can see that (1) for problems with the dimension $n = 10$, the best choice is $m = 2$; (2) for problems with dimension $n = 20$ and 50, the best choice is $m = 3$; (3) $m = 2$ works best on problems with $n = 10$; (4) $m = 3$ works best on problems with $n = 20$; (5) $m = 3$ works best on problems with $n = 20$. We can conclude that the optimal choice of m will increase very slowly with the increase of problem dimension.

5.3 *An implementation issue*

Although modified LDSE has been proved to be global convergent, it may fail to find the global optimum within a limited *nfe*. If you need a high reliability for your optimization result, then you can follow the result from Section 4.2. Modified LDSE can find the global optimum with probability not less than $1 - (T_0/T)^{\lambda}$, where T_0 is the upper bounds of its first passage time, T is the maximum number of generations, λ is the number which modified LDSE is executed. Usually, the exact T_0 is unknown, but it can be estimated by $T_0 \approx nfe/N$. Take the problem ACK ($n = 20$) as an example (see Table 1), $T_0 \approx nfe/N = 28750/40 \approx 720$. Therefore, we can get the global minimum of ACK with probability not less than $1 - 10^{-6}$ if we execute modified LDSE 10 times independently with the fixed number of generations $T = 4 \cdot T_0 = 2880$, or equivalently, if we execute modified LDSE 20 times with $T = 2 \cdot T_0 = 1440$. Note that the probability $1 - 10^{-6}$ here is an upper boundary, the actual probability will be much closer to 1.

6. Conclusion

We have introduced three techniques, including LDR, NS and VD to improve the performance of LDSE. LDR makes LDSE capable of finding the global optimum with a small population. NS can enhance LDSE's local search ability. VD can help LDSE make better use of recent information. The performance of modified versions of LDSE with the three techniques is assessed using two groups of test functions. Numerical results show that these techniques improve not only LDSE's efficiency, but also its reliability. Modified LDSE outperforms basic LDSE, and it performs better than DERL considerably in most cases.

We have also analysed the convergence properties of LDSEs by means of finite Markov chains. It shows that basic LDSE might fail to converge, but modified LDSE (with the above three techniques) will converge for any initial population. In addition, a formula for estimating the convergence speed of modified LDSE is provided.

Although numerical results show that modified versions of LDSE work better than basic LDSE, the modifications are still heuristic. Their working mechanism cannot be explained completely with the presented theoretical results and discussions. However, the idea of the three techniques should be universal, and it could also be used to improve the performance of other real-coded EAs. More comparisons of modified LDSE and other GO algorithms should be carried out to give modified LDSE a more comprehensive evaluation. These topics are left for our future research.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (Grant No. 90916028) and a Grant-in-Aid for 21st Century COE Frontiers of Computational Science in Japan.

References

[1] M.M. Ali, C. Khompatraporn, and Z.B. Zabinsky, *A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems*, J. Global Optim. 31 (2005), pp. 635–672.
[2] U.K. Chakraborty, D. Kalyanmoy, and M. Chakraborty, *Analysis of selection algorithms: a Markov chain approach*, Evol. Comput. 4 (1996), pp. 133–167.
[3] T.E. Davis and J.C. Principe, *A Markov chain framework for the simple genetic algorithm*, Evol. Comput. 1 (1993), pp. 269–288.
[4] D.B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, 3rd ed., Wiley-IEEE Press, Piscataway, NJ, 2006.
[5] L.J. Fogel, A.J. Owens, and M.J. Walsh, *Artificial intelligence through a simulation of evolution*, Biophysics and Cybernetic Systems: *Proc. 2nd Cybernetic Sciences Symposium*, M. Maxfield, A. Callahan and L.J. Fogel, eds., Washington DC: Spartan Books, pp. 131–155, 1965.
[6] S. Garcia, A. Fernandez, J. Luengo, and F. Herrera, *A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability*, Soft Comput. 13 (2009), pp. 959–977.
[7] M.H. Gutknecht, *A brief introduction to krylov space methods for solving linear systems*, Frontiers of Computational Science, Springer Berlin Heidelberg, (2007), pp. 53–62.
[8] A. Hedar and M. Fukushima, *Minimizing multimodal functions by simplex coding genetic algorithm*, Optim. Method Softw. 18 (2003), pp. 265–282.
[9] J.H. Holland, *Adaption in Natural and Artificial Systems*, MIT Press, Cambridge, MA, 1992.
[10] P. Kaelo and M.M. Ali, *A numerical study of some modified differential evolution algorithms*, Eur. J. Oper. Res. 169 (2006), pp. 1176–1181.
[11] J. Kennedy and R. Eberhart, *Particle swarm optimization*, Proc, IEEE Int'l. Conf. on Neural Networks, Vol. 4, 1995, pp. 1942–1948.
[12] C.T. Luo and B. Yu, *Low dimensional simplex evolution: a hybrid heuristic for global optimization*, Global Optim. (DOI:10.1007/s10898-011-96781).
[13] C.T. Luo and B. Yu, *Solving min UR problem by triangle evolution algorithm with archiving and Niche techniques*, J. Inform. Comput. Sci. 4(1) (2007), pp. 195–203.
[14] *Global Optimization Toolbox 3 Users Guide*, The MathWorks, Inc., Natick, MA, 2010.
[15] C.D. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, PA, 2000.
[16] R. Poli and N.F. McPhee, *General schema theory for genetic programming with subtree-swapping crossover: Part II*, Evol. Comput. 11 (2003), pp. 169–206.
[17] I. Rechenberg, *Evolutions strategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*, Frommann-Holzboog Verlag, Stuttgart, 1973.
[18] J.M. Renders and H. Bersini, *Hybridizing genetic algorithms with hill-climbing methods for global optimization: Two possible ways*, Proc. 1st IEEE Conf. Evolutionary Computation. Vol. 1, 1994, pp. 312–317.
[19] G. Rudolph, *Convergence analysis of canonical genetic algorithms*, IEEE Trans. Neural Network 5 (1994), pp. 96–101.
[20] L.M. Schmitt, *Theory of genetic algorithms II: models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling*, Theor. Comput. Sci. 310 (2004), pp. 181–231.

[21] R. Storn and K. Price, *DE-a simple and efficient heuristic for global optimization over continuous space*, J. Global Optim. 11 (1997), pp. 341–359.
[22] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, *Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization*, Technical Report, Nanyang Technological University, Singapore, 2005.
[23] B. Yu, C.T. Luo, and Y. Zhang, *Studies on Complicated System of Inequalities with Possible Inconsistency*, Frontiers Science Series, 49 (2007), pp. 147–148.