Soft Computing manuscript No.
(will be inserted by the editor)

Accelerating Differential Evolution Based on a Subset-to-Subset Survivor Selection Operator

Jinglei Guo · Zhijian Li · Shengxiang Yang

Received: December 4, 2017; Revised: January 10, 2018

Abstract Differential evolution (DE) is one of the most powerful and effective evolutionary algorithms for solving global optimization problems. However, just like all other metaheuristics, DE also has some drawbacks, such as slow and/or premature convergence. This paper proposes a new subset-to-subset selection operator to improve the convergence performance of DE by randomly dividing target and trial populations into several subsets and employing the ranking-based selection operator among corresponding subsets. The proposed framework gives more survival opportunities to trial vectors with better objective function values. Experimental results show that the proposed method significantly improves the performance of the original DE algorithm and several state-of-the-art DE variants on a series of benchmark functions.

Keywords Differential evolution · global optimization · subset-to-subset survivor selection · convergence

1 Introduction

Differential evolution (DE), which was first proposed by Storn and Price (Price et al., 2006; Storn and Price, 1995), has become one of the most powerful stochastic real-parameter optimization algorithms. Similar to other evolutionary algorithms (EAs), DE employs three evolutionary operations: mutation, crossover and selection to evolve the population towards the global optimum. However, unlike traditional EAs, DE employs the difference of parameter vectors to explore different regions of the search space. In DE, a parent vector from the current population is called the *target* vector, a *donor* vector is obtained through the differential mutation operator, and finally, an offspring is formed by recombining the donor vector with the target vector, which is called the *trial* vector. In addition, DE employs a one-to-one replacement scheme during the selection process.

Several DE variants have secured high ranks in a number of IEEE Congress on Evolutionary Computation (IEEE CEC) competitions, e.g., the 2006 IEEE CEC Competition on Constrained Real Parameter Optimization (ranked the first), the 2007 IEEE CEC Competition on Multi-objective Optimization (ranked the second), the 2008 IEEE CEC Competition on Large Scale Global Optimization (ranked the third), the 2009 IEEE CEC Competition on Multi-objective Optimization (the first rank was taken by a DE-based multi-objective optimization EA with decomposition (MOEA/D) for unconstrained problems), and the 2009 IEEE CEC Competition on Evolutionary Computation in Dynamic and Uncertain Environments (ranked the first) (Das and Suganthan, 2011). DE has been successfully applied in real-world complex engineering optimization areas during the past two decades, such as electrical power systems (Cai et al., 2008), electromagnetism (Qing, 2006), microwave engineering (Massa et al., 2006), control systems (Cruz et al., 2003), manufacturing industry (Yildiz, 2013), etc. A survey on the state-of-the-art research for DE can be found in (Das et al., 2016).

In the classical DE algorithm, one-to-one survivor selection is employed to determine whether the target vector or the trial vector survives to the next generation. Comparing each trial vector to the best performing vector at the same index retains not only the best vector at each index, but also the very best-so-far solution at any index. This selection strategy, however, can slow down DE's speed of con-

Jinglei Guo · Zhijian Li
School of Computer Science, Central China Normal University, Wuhan 430079, China E-mail: jingleiguo@mail.ccnu.edu.cn, lzj1769@163.com
Shengxiang Yang
Centre for Computational Intelligence (CCI), School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, U.K. E-mail: syang@dmu.ac.uk

vergence. For example, a trial vector that is better than most of the current population solutions will be rejected if its target vector is even better. Motivated by these findings, we propose a subset-to-subset (STS) selection scheme to select offspring for survival instead of the one-to-one offspring selection scheme in this paper. In the proposed STS selection scheme, the target and trial populations are randomly divided into a number of subsets. In the corresponding subsets, vectors are selected according to their fitness. By selecting the best vectors from corresponding subsets, this scheme tries to keep better trial vectors into the next generation and hence accelerates the convergence toward the optimal solution(s). By the selection between the corresponding subset of parent population and child population, this framework promotes efficient exploitation without substantially diminishing the exploration capability of population. We incorporate this selection framework to the classical DE and a number of important DE variants and carry out experiments to observe the performance gains.

The rest of this paper is organized as follows. Section 2 briefly introduces the classical DE algorithm and surveys related works on improving DE's performance. In Section 3, the proposed STS selection scheme is described in detail. Experimental results on the well-known 13 high-dimensional benchmark functions (Brest et al., 2006; Brest and Maucec, 2009; Wang et al., 2013; Yao et al., 1999; Zhang and Sanderson, 2009) (denoted as the basic test bed in this paper) and benchmark problems used for 2014 IEEE CEC Competition on Single Objective Real-parameter Numerical Optimization (denoted as the 2014 IEEE CEC benchmark functions in this paper) are reported in Section 4. Finally, Section 5 concludes this paper.

2 Classical DE Algorithm

The framework of the classical DE algorithm mainly consists of four parts: initialization, mutation, crossover and selection, which are described respectively below.

2.1 Initialization

DE starts with a population of NP D-dimensional parameter vectors in the search space, where NP is the population size and D is the problem dimensionality. The *j*th component of the *i*th vector may be initialized as:

$$x_{i,j}^0 = x_{j,min} + rand() \cdot (x_{j,max} - x_{j,min}) \tag{1}$$

where $rand()$ generates a uniformly distributed random number in the range $[0, 1]$, $x_{j,min}$ and $x_{j,max}$ are the minimum and maximum bounds of the *j*th component of a vector, respectively.

2.2 Mutation

After initialization, for each vector X_i^G at generation G, a mutant vector V_i^G is generated by a certain mutation strategy. The most frequently used mutation strategies are defined as follows:

- DE/rand/1:

$$V_i^G = X_{r1}^G + F \cdot (X_{r2}^G - X_{r3}^G) \tag{2}$$

- DE/best/1:

$$V_i^G = X_{best}^G + F \cdot (X_{r1}^G - X_{r2}^G) \tag{3}$$

- DE/current-to-best/1:

$$V_i^G = X_i^G + F \cdot (X_{best}^G - X_i^G) + F \cdot (X_{r1}^G - X_{r2}^G) \tag{4}$$

- DE/rand/2:

$$V_i^G = X_i^G + F \cdot (X_{r2}^G - X_{r3}^G) + F \cdot (X_{r4}^G - X_{r5}^G) \tag{5}$$

- DE/best/2:

$$V_i^G = X_{best}^G + F \cdot (X_{r1}^G - X_{r2}^G) + F \cdot (X_{r3}^G - X_{r4}^G) \tag{6}$$

where F is the scaling factor and the indices r_1, r_2, r_3, r_4 and r_5 are mutually different random integers selected from $\{1, 2, \cdots, NP\}$, which are different from the base index i. X_{best}^G is the best vector in terms of fitness value at the current population G.

2.3 Crossover

After mutation, DE generates a trial vector $U_i^G = \{u_{i,1}^G, \cdots, u_{i,D}^G\}$ by recombining $X_i^G = \{x_{i,1}^G, \cdots, x_{i,D}^G\}$ and $V_i^G = \{v_{i,1}^G, \cdots, v_{i,D}^G\}$ as follows:

$$u_{i,j}^G = \begin{cases} v_{i,j}^G & if\ (rand_{i,j}(0,1) \leq Cr\ or\ j = j_{rand}) \\ x_{i,j}^G & otherwise \end{cases} \tag{7}$$

where Cr is the crossover rate, $rand_{i,j}(0, 1)$ generates a random number in $[0, 1]$, and $j_{rand} \in \{1, 2, \ldots, D\}$ is a randomly chosen index, which ensures that U_i^G gets at least one parameter from V_i^G.

2.4 Selection

Finally, a greedy selection operator is used to determine whether the target or trial vector survives to the next generation, which is described as follows:

$$X_i^{G+1} = \begin{cases} U_i^G & if\ f(U_i^G) \leq f(X_i^G) \\ X_i^G & otherwise \end{cases} \tag{8}$$

where $f(X)$ is the objective function (without lose of generality, we assume minimization problems in this paper).

3 Related Work

In the past twenty years, DE has attracted many researchers' attention and many DE variants are developed. The current study of DE focus on three aspects: 1) adapting the control parameters; 2) improving trial vector generation strategies; 3) improving the selection operation in the mutation and survival stages.

3.1 Adapting the Control Parameter Settings

Brest *et al.* (Brest et al., 2006) proposed a self-adaptation scheme for the DE control parameters by encoding parameters F and Cr into each individual and adjusting them during the evolutionary process. Zhang and Sanderson (Zhang and Sanderson, 2009) proposed an adaptive DE with an optional external archive (JADE), where the scaling factor F and the crossover rate Cr are generated according to a normal distribution and a Cauchy distribution, respectively. In the self-adaptive DE (SaDE) (Qin et al., 2009), the parameter F is approximated by a normal distribution with the mean value 0.5 and standard deviation 0.3, denoted by $N(0.5, 0.3)$, and Cr is adjusted gradually for a given problem according to the previous Cr values that have generated trial vectors successfully entered the next generation. In addition, four effective trial vector generation strategies are chosen to constitute a strategy candidate pool. A mutation strategy is selected from the strategy candidate pool according to the probability learned from its success rate in generating better individuals within a certain number of previous generations, called the learning period. FiADE, introduced in (Ghosh et al., 2011), is a very simple yet very efficient adaptation technique for tuning both F and CR, during the run, without any user intervention. The parameter adaptation strategy is based on the objective function values (fitness values) of individuals in the DE population. The presented results showed that the performance of FiADE is very competitive with the best-known DE variants. The Success-History based Adaptive DE (SHADE) (Tanabe and Fukunaga, 2013) is an improved version of JADE (Zhang and Sanderson, 2009), which uses a different parameter adaption mechanism based on the history of success. In (Tanabe and Fukunaga, 2014), Tanabe and Fukunaga proposed a modified SHADE algorithm, denoted L-SHADE, which incorporated a Linear Population Size Reduction (LPSR) scheme into SHADE. LPSR is a simple deterministic population resizing method that continuously reduces the population size in accordance with a linear function. The experimental results over the 2014 IEEE CEC benchmark functions show that L-SHADE is quite competitive in comparison with state-of-the-art EAs. jSO (Brest et al., 2017) is an improved variant of the LSHADE algorithm with a new weighted parameter F_w to adjust the vector X^G_{pbest}. jSO ranked 2nd place in 2017 IEEE CEC competition on Bound Constrained Benchmark Set.

3.2 Improving Trial Vector Generation Strategies

In (Rahnamayan et al., 2008), the opposition-based DE (ODE) employs opposition-based learning (OBL) for population initialization and generation to accelerate DE's convergence speed. Das *et al.* (Das et al., 2009) proposed a family of improved variants of the DE/target-to-best/1/bin scheme using the concept of the neighbourhood of each population member. In (Wang et al., 2011), the composite DE (CoDE) was proposed to generate a trial vector by using three trial vector generation strategies and three control parameter settings. Experimental results showed that CoDE is very competitive on the 2005 IEEE CEC benchmark functions. An ensemble of mutation strategies and control parameters with DE, denoted EPSDE, was proposed in (Mallipeddi et al., 2011). The performance of EPSDE was evaluated on a set of bound-constrained problems and competitive with conventional DE and several state-of-the-art parameter adaptive DE variants. In (Wang et al., 2013), a Gaussian bare-bones DE (GBDE) and its modified version (MGBDE) were proposed, which are almost parameter free. THDE (Peng and Wu, 2015) employs Taguchi method to generate trial vectors based on orthogonal array. In (Yu et al., 2014), the authors proposed an adaptive DE (ADE) algorithm with a new mutation strategy DE/lbest/1, which is beneficial to the balance between fast convergence and population diversity, and a two-level adaptive parameter control scheme. DBDE (Peng et al., 2016) introduces the new dichotomous mutation and dichotomous crossover to solve 0-1 knapsack problem and multidimensional knapsack problem. Peng (Peng et al., 2017) proposed a DE/neighbor/1 mutation strategy in which one solution is selected from the best one in the neighbors. Wang (Wang et al., 2016) eliminated the restrained condition(mutually different indices) in mutation strategy and suggested DE without restrained condition may be useful in high dimensional search space. TSDE (Liu et al., 2016) utilizes the different trial vector generation strategies combination at different stages.

More recently, much effort has been made to utilize the cumulative correlation information already existing in the evolutionary process. Li *et al.* (Li et al., 2015) proposed a predictive approach, denoted DEEP, to the reproduction mechanism of new individuals for DE algorithms. The DEEP framework offers advantages of both a distributed model (DM) and a centralized model (CM), hence substantially enhances DE's performance. Two DEEP variants were developed and illustrated in the paper. In (Wang et al., 2014, 2016), Wang utilized the population distribution information to establish an Eigen coordinate system and generated trial vectors in the Eigen coordinate system with a pre-

defined probability. In (Segura et al., 2015), the authors provided a better insight into the reasons of the "curse of dimensionality" and proposed techniques to alleviate this problem. Computational results on a large set of scalable problems with various complexities showed that the new proposal is much better than the original DE scheme.

3.3 The Selection Operation in DE

There are primarily two stages in the evolutionary process of a DE algorithm, where selection can be applied to a population, i.e., *parent selection*, which decides which vectors will undergo recombination, and *survival selection*, which chooses the vectors from the current target vectors and trial vectors into the next generation (Price et al., 2006).

In (Gong and Cai, 2013), Gong and Cai proposed a ranking-based mutation operator for DE, where some of the parents in the mutation operation are proportionally selected according to their rankings in the current population. The higher ranking a parent obtains, the more opportunity it will be selected for mutation. Experimental results indicate that the ranking-based mutation operator is able to enhance the performance of the original DE algorithm and other advanced DE algorithms. In (Epitropakis et al., 2011), the selection of parents is based on the distance between the solutions in the current population. The solutions with a small distance have higher probabilities to be selected as parents. In (Guo et al., 2014), an effective and efficient successful-parent-selecting framework was proposed to improve the performance of DE by providing alternatives for the selection of parents during mutation and crossover. The proposed method adapts the selection of parents by storing successful solutions into an archive, and the parents are selected from the archive when a solution has not been continuously updated for a certain number of times.

In addition to improving the parent selection operator, the following approaches focus on survival selection. In (Segura et al., 2013), the authors proposed a new survival selection scheme (DCN-THR-REF) to avoid the loss of diversity, in which the individual is evaluated by the fitness and the distance to the closest neighbour (DCN) simultaneously. Certainly, the individuals with better fitness and larger DCN are survived to the next generation by the diversity-based multi-objective DCN-THR-REF method. Results indicated that the proposed approach provides several advantages in avoiding premature convergence. However, DCN-THR-REF method produces a reduction in the convergence speed and expensive computation cost for finding the better individuals of two objectives. In (Tagawa, 2009), Tagawa proposed five survival selection methods: Family Survival Selection (CF), where the worse target vector is replaced by the corresponding trial vector immediately; Worst Survival Selection (CW), where the trial vector is always compared with the worst individual in the population; Absolute Survival Selection (CA), where the worst individual in the population is replaced by the trial vector without the comparison; Closest Survival Selection (CC), where the trial vector is always compared with the closest individual with it in the population; Hybrid Survival Selection (CH), where the combination of the closest survival selection and the worst survival selection. Authors pointed out that CW enhances the convergence speed of DE, while CC improves the robustness from the experiment report.

4 Subset-to-Subset (STS) Survival Selection

In this section, we first investigate the impact of one-to-one survival selection on DE in Section 4.1, then present the proposed STS operator in Section 4.2, and finally give the computational cost of the STS operator in Section 4.3.

4.1 Impact of the One-to-One Survivor Selection Operator

In the classical DE algorithm, the survival selection determines whether the target or trial vector survives to the next generation. The trial vector U_i^G replaces the target vector X_i^G if it has an equal or better objective function value; otherwise, the target vector retains its place in the population for at least one more generation. The best performing vector at the ith position is just the ith vector in the current population, i.e., the target vector X_i^G (Price et al., 2006). However, this selection scheme has two shortcomings. The first one is that a trial vector with a better objective function value than most of other current population members will be rejected if its corresponding target vector is even better. The other one, in contrast, is that a trial vector with a poor objective function value will survive to the next generation if its corresponding target vector is even worse.

In order to clearly show the above phenomenon, we define the variable rtv^G as the *number of rejected trial vectors* in the Gth generation as follows:

$$rtv^G = \sum_{i=1}^{NP} r_i^G \tag{9}$$

where r_i^G is an indicator which shows whether the ith trial vector is rejected or not, i.e.:

$$r_i^G = \begin{cases} 1, & if\ f(U_i^G) > f(X_i^G) \\ 0, & otherwise \end{cases} \tag{10}$$

For each rejected trial vector U_i^G in the Gth generation, the number of vectors with a worse objective function value than it in the current population is called its *rejection error*

index (REI), denoted rei_i^G, which can be calculated by:

$$rei_i^G = \sum_{j=1}^{NP} e_{ij}^G \tag{11}$$

where e_{ij}^G is an indicator which shows whether the ith trial vector is rejected unfairly or not in comparison with vector j in the current population, i.e.:

$$e_{ij}^G = \begin{cases} 1, & if\ f(U_i^G) < f(X_j^G) \quad j \in \{1, 2, \cdots, NP\} \\ 0, & otherwise \end{cases} \tag{12}$$

The mean value of REI at generation G is defined as follows:

$$rei^G = \big(\sum_{i=1}^{rtv^G} rei_i^G \big) / rtv^G \tag{13}$$

If $rtv^G = 0$, which means that all trial vectors are accepted, we set $rei^G = 0$ directly.

A higher value of rei^G indicates that some trial vectors with better objective function values than most other target vectors in the Gth population are rejected inappropriately since their target vectors are better. If we can reduce the value of rei^G, the convergence of DE is expected to be accelerated and the performance of DE is expected to be improved. This motivates our STS selection operator which aims to appropriately accept trial vectors with better objective function values and reject trial vectors with worse objective function values as many as possible by dividing the target and trial populations into a number of subsets and employing ranking-based selection in corresponding subsets.

4.2 Description of the STS Survivor Selection Operator

In the STS selection scheme, the vectors are organized in a ring topology with respect to their indices, which is motivated by (Das et al., 2009), and the number of subsets to be formed depends on the subset size, denoted SS, which is given in advance. The subsets are formed as follows. Supposing that we have a target population $P^G = \{X_1^G, X_2^G, \cdots, X_{NP}^G\}$ and a trial population $TP^G = \{U_1^G, U_2^G, \cdots, U_{NP}^G\}$ in the Gth generation, we first randomly select an index $i \in \{1, 2, \cdots, NP\}$ as the starting point which is updated every new generation and then divide both P^G and TP^G into n subsets immediately, denoted as TAS_j^G and TRS_j^G ($j \in \{1, 2, \cdots, n\}$), respectively. From each pair of subsets TAS_j^G and TRS_j^G ($j \in \{1, 2, \cdots, n\}$), we select the best SS vectors to survive into the next generation. Obviously, the one-to-one selection operator of the classical DE algorithm is a special case of STS with $SS = 1$. The pseudo-code of the proposed STS selection is shown in Algorithm 1.

Algorithm 1 Subset-to-subset (STS) selection

1: Input: the subset size SS
   {/* Calculate the number of subsets: n */}
2: **if** $NP\%SS == 0$ **then**
3: $n := NP/SS$
4: **else**
5: $n := NP/SS + 1$
6: **end if**
7: $i := rand() \times NP$ {/* Select randomly an integer in $\{1, 2, \ldots, NP\}$ as the starting index */}
8: **for** $j := 1$ **to** n **do**
9: $NR := NP - j \times SS$ {/* Calculate the number of rest vectors which have not undergone the survivor selection yet in the target population */}
10: **if** $NR > SS$ **then**
11: Take SS vectors from the target population P^G and trial population TP^G as subsets TAS_j^G and TRS_j^G respectively starting from index i
12: Select the best SS vectors from the joint set $\{TAS_j^G, TRS_j^G\}$ as P_j^{G+1}
13: $i := (i + SS)\%NP$
14: **else**
15: Take NR vectors from the target population P^G and the trial population TP^G as subsets TAS_j^G and TRS_j^G respectively starting from the index i
16: Select the best NR vectors from the joint set $\{TAS_j^G, TRS_j^G\}$ as P_j^{G+1}
17: **end if**
18: **end for**
19: $P^{G+1} := \sum_{j:=1}^{n} P_j^{G+1}$

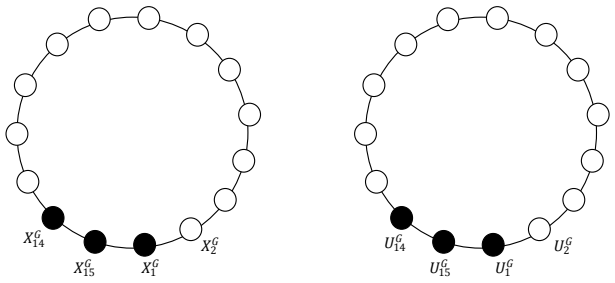

Fig. 1 Ring topology of 15 vectors. The left and right figures are the target and trial populations respectively. The black points indicate the first pair of corresponding subsets, where the starting index i is 14 and the subset size SS is 3.

Fig. 1 illustrates an example of dividing the population into a number of subsets, where the population size is 15, the subset size SS is 3, and the starting index is 14. The proposed method selects the best 3 vectors $\{X_{14}^{G+1}, X_{15}^{G+1}, X_1^{G+1}\}$ from the set $\{X_{14}^G, X_{15}^G, X_1^G, U_{14}^G, U_{15}^G, U_1^G\}$ to the next generation and performs a similar operation for other corresponding subsets.

In the one-to-one selection of the classical DE algorithm, a trial vector with a better objective value than other target vectors may be rejected unfairly. In the STS selection, a better trial vector will have a higher opportunity to be accepted and hence the performance of DE may be im-

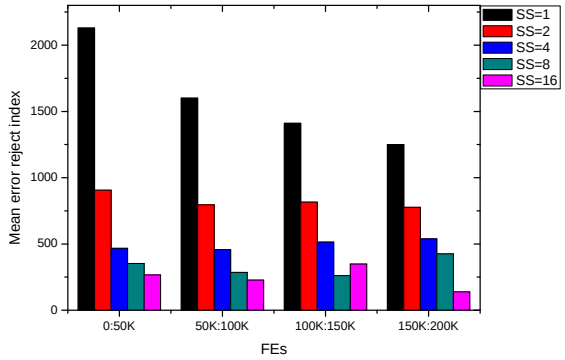

Fig. 2 The accumulation of mean rejection error index of running DE/rand/1/bin on the 30-dimensional Schwefel 2.21 function, where $NP = 100$ and $SS = 1$, $SS = 2$, $SS = 4$, $SS = 8$ and $SS = 16$, respectively.

proved. Fig. 2 shows the accumulation of mean REI of running DE/rand/1 with the proposed STS operator on the 30-dimensional Schwefel 2.21 function over 5000 generations, where $NP = 100$ and SS is set to 1, 2, 4, 8 and 16, respectively. As shown in Fig. 2, the value of SS has a significant impact on the mean REI (rei^G), where a large SS reduces the mean REI. In this paper, a DE algorithm with the STS scheme integrated will be denoted with the prefix "STS". For example, the classical DE algorithm with the STS scheme will be denoted as STS-DE.

To the best of our knowledge, DEGL (Das et al., 2009) was the first attempt to utilize the concept of the neighbourhood subset to provide an improved variants of the DE/cuttent-to-best/1. Our proposed STS scheme differs from DEGL in the following aspects.

- The concept of subset in this paper is used to improve the selection of survivors, while the concept of subset (neighbourhood) in DEGL is applied to improve the parents selection.
- The division method is different. The subsets in this paper are disjoint, while the subsets in DEGL are overlapping.

4.3 The Computational Cost of the STS Selection Operator

In the proposed STS selection, target and trial populations are divided into a number of subsets. In Algorithm 1, the computational complexity of the STS selection depends on the size of subsets (SS). To select the best SS individuals from the corresponding subsets, we need $2 \cdot SS \cdot log(2 \cdot SS)$ comparisons. Thus, the computational cost of the STS selection is:

$$Cost_{STS} = n \cdot 2 \cdot SS \cdot log(2 \cdot SS) = 2 \cdot NP \cdot log(2 \cdot SS) \quad (14)$$

where n is the number of subsets and NP is the population size. Obviously, the computational cost of the STS selection is a monotonically increasing function of SS. However, the computational cost of algorithms is almost spent on the calculation of the fitness function rather than the selection operator, hence the STS selection operator does not significantly increase the computational cost.

5 Experimental Study

5.1 Test Functions

In order to test the effectiveness of the STS operator on diverse test functions and analyze the integrated DE algorithm with the STS operator, we choose two groups of benchmark test functions. Group A includes high-dimensional functions (basic unimodal and multimodal functions), while Group B includes shifted and/or rotated functions (unimodal, multimodal, hybrid and composition functions).

5.1.1 Group A (Unimodal and Multimodal Functions)

As a starting point, the basic test bed is used to compare STS-DE with DE. This basic test bed includes 13 well-known benchmark functions, which have been widely used in the literature (Brest et al., 2006; Brest and Maucec, 2009; Wang et al., 2013; Yao et al., 1999; Zhang and Sanderson, 2009). The detailed description of these benchmark functions is shown in Table 1. Here, f_1 -f_5 are continuous unimodal functions, of which f_5 has a narrow valley from the perceived local optimum to the global optimum, f_6 is a discontinuous step function, f_7 is the noisy quartic function, and f_8-f_{13} are multimodal functions with many local minima.

5.1.2 Group B (Shifted and/or Rotated Functions)

This group consists of the rotated and/or shifted benchmark functions used for the 2014 IEEE CEC Competition on Single Objective Real-parameter Numerical Optimization (Liang et al., 2013). They are defined as:

$$F(x) = cf(M(x - o)) \quad (15)$$

where M is the orthogonal matrix and o is the shifted vector. Based on the characteristics of the basic functions, the 2014 IEEE CEC benchmark functions can be divided into four classes: Functions $cf_1 - cf_3$ are unimodal, functions $cf_4 - cf_{16}$ are simple multimodal functions, functions $cf_{17} - cf_{22}$ are hybrid, and functions $cf_{23} - cf_{30}$ are composition functions with a huge number of local minima. A thorough description of this suit is provided in (Liang et al., 2013).

Table 1 The basic test bed of 13 well-known benchmark functions

Name	Test functions	Initial range
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$
Schwefel 2.22	$f_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} x_i$	$[-10, 10]^D$
Schwefel 1.2	$f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	$[-100, 100]^D$
Schwefel 2.21	$f_4(x) = max_i\{\lvert x_i \rvert\}$	$[-100, 100]^D$
Rosenbrock	$f_5(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]^D$
Step	$f_6(x) = \sum_{i=1}^{D} \lfloor x_i + 0.5 \rfloor^2$	$[-100, 100]^D$
Quartic with noise	$f_7(x) = \sum_{i=1}^{D} i \cdot x_i^4 + random[0, 1)$	$[-1.28, 1.25]^D$
Schwefel 2.6	$f_8(x) = \sum_{i=1}^{D} -x_i \cdot sin(\sqrt{\lvert x_i \rvert}) + D \cdot 418.9829$	$[-500, 500]^D$
Rastrigin	$f_9(x) = \sum_{i=1}^{D} [x_i^2 - 10cos2\pi x_i + 10]$	$[-5.12, 5.12]^D$
Ackley	$f_{10}(x) = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D}\sum_{i=1}^{D} cos(2\pi x_i)) + 20 + e$	$[-32, 32]^D$
Griewank	$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{D} (x_i)^2 - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]^D$
Penalized 1	$f_{12}(x) = \frac{\pi}{D}\{\sum_{i}^{D-1} (y_i - 1)^2[1 + sin(\pi y_{i+1})] + (y_D - 1)^2 + 10sin^2(\pi y_1)\}$ $+ \sum_{i=1}^{D} u(x_i, 10, 100, 4), y_i = 1 + \frac{1}{4}(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < a \end{cases}$	$[-50, 50]^D$
Penalized 2	$f_{13}(x) = 0.1\{sin^2(3\pi x_1) + \sum_{i=1}^{D=1} (x_i - 1)^2[1 + sin^2(3\pi x_{i+1})]$ $+(x_D - 1)^2[1 + sin^2(2\pi x_D)]\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$

5.2 Relationship Between the Population Size and Subset Size

A new parameter, the subset size (SS), i.e., the number of individuals in a subset, is introduced in the proposed STS selection operator. The value of SS affects the convergence speed and solution accuracy. Obviously, a large SS accelerates convergence by providing more chances for trial vectors with better objective function values. However, the integrated DE algorithm may lose population diversity rapidly and suffer from the problem of premature convergence. On the other hand, a small SS retains population diversity, but the integrated DE algorithm may converge slowly. In this section, we would like to find out the proper subset size for the STS operator to be integrated with the DE algorithm. In addition, it is necessary to analyse the relationship between SS and the population size NP.

Table 1 lists the well-known benchmark functions, on which we conduct experiments to compare STS-DE with DE. The specific parameter settings in the experiments are listed as follows: 1) The population size NP is set to 100, 150 and 200, respectively; 2) $F = 0.5$ and $CR = 0.9$; 3) The mutation scheme is DE/rand/1; 4) An algorithm stops when the number of fitness evaluations FEs reaches the allowed maximum value, denoted MAX_FEs, which is set to $2.0E{+}05$; 5) Each algorithm is run 30 times per function.

The results achieved by DE/rand/1 and STS-DE/rand/1 are summarized in Table 2. The best mean error for each function within a given population size is shown in the underline format. To highlight the algorithm with the best parameter setting, the overall best mean error for each function across different population sizes is also shown in **boldface**. For example, when $NP = 150$, the best mean error for f_3 is obtained by STS-DE/rand/1 with $SS = 16$, which is shown as $\underline{1.34e-03}$ in Table 2. But, for f_3, the overall best mean error is obtained by STS-DE/rand/1 with $NP = 100$ and $SS = 16$, which is shown boldfaced as well as underlined as **$\underline{\text{1.68e-09}}$**.

When $SS = 1$, the STS variant is the classical DE/rand/1. Based on the results, the STS variant with $SS > 1$ achieves better results than DE/rand/1 on 9, 11 and 12 functions when the population size (NP) is 100, 150 and 200, respectively. It is obvious that the STS operator has greater effect on convergence accuracy when the population size is increasing. The reason is that the original algorithm is under strong selection pressure when the population size is small. Among the 13 testing functions, DE/rand/1 and STS-DE/rand/1 both get the best results on f_6, while STS-DE/rand/1 obtains the best results for the rest functions. In these functions, f_6 is a step and discontinuous function, moreover all testing algorithms reach the best result on this function. Under the condition $NP = 100$, every STS-DE/rand/1 algorithm with $SS > 1$ obtains better results on 9 functions, and worse results on 2 functions from Table 2. Meanwhile, STS-DE/rand/1 obtains the best results on 1, 5, 5, 8, and 5 function(s) with SS set to 2, 4, 8, 16 and 100, respectively. Considering the computational cost, it is a wise choice to set SS in the interval $[4, 8]$ when $NP = 100$. In the testing group of $NP = 150$, every STS-DE/rand/1 with $SS > 1$ wins on 11 functions and does not lose on any functions. Meanwhile, STS-DE/rand/1 obtains the best results on 1, 2, 6, 6 and 3 function(s) when SS was set to 2, 4, 8, 16 and 150, respectively. So, it is reasonable to set SS in the interval $[8, 16]$ for $NP = 150$. When $NP = 200$, it is advisable to set $SS = 16$ because the STS-DE/rand/1 with

Table 2 The error values for different population sizes at $D = 30$, after 200,000 fitness evaluations

	$NP = 100$											
	$SS = 1$		$SS = 2$		$SS = 4$		$SS = 8$		$SS = 16$		$SS = NP$	
	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.
f_1	5.51e-20	4.69e-20	4.93e-25(+)	4.80e-25	5.43e-42(+)	4.62e-42	1.35e-42(+)	2.20e-42	2.62e-43(+)	2.31e-43	**1.25e-59**(+)	2.67e-59
f_2	4.86e-10	2.25e-10	2.01e-12(+)	1.20e-12	1.16e-20(+)	1.05e-20	6.59e-17(+)	3.36e-16	1.64e-17(+)	8.39e-17	**1.52e-31**(+)	2.10e-31
f_3	9.63e-03	7.01e-03	1.83e-04(+)	1.39e-04	4.47e-09(+)	3.83e-09	1.79e-09(+)	1.94e-09	**1.68e-09**(+)	2.61e-09	2.10e-05(+)	1.82e-05
f_4	7.06e-02	1.43e-01	4.02e-01(−)	6.23e-01	5.65e-01(−)	1.06e+00	1.30e+00(−)	2.11e+00	5.76e-01(−)	8.99e-01	1.23e+00(−)	9.90e-01
f_5	1.13e+01	1.04e+00	1.63e+01(−)	1.74e+00	1.53e+01(−)	2.50e+00	1.81e+01(−)	1.85e+00	1.84e+01(−)	1.38e+00	2.11e+01(−)	3.00e-01
f_6	**0.00e+00**	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00
f_7	6.27e-03	1.65e-03	5.43e-03(+)	1.45e-03	**2.96e-03**(+)	7.85e-04	2.97e-03(+)	9.31e-04	3.14e-03(+)	7.83e-04	5.23e-03(+)	1.14e-03
f_8	7.11e+03	3.12e+02	6.60e+03(+)	6.19e+02	6.66e+03(+)	6.45e+02	6.55e+03(+)	4.88e+02	**6.49e+03**(+)	7.63e+02	6.98e+03(+)	4.00e+02
f_9	1.63e+02	2.09e+01	1.46e+02(+)	1.92e+01	1.02e+02(+)	2.53e+01	1.06e+02(+)	3.35e+01	**9.78e+01**(+)	3.27e+01	1.14e+02(+)	1.40e+01
f_{10}	6.00e-11	2.11e-11	2.50e-13(+)	1.31e-13	**4.00e-15**(+)	1.58e-30	**4.00e-15**(+)	1.58e-30	**4.00e-15**(+)	1.58e-30	**4.00e-15**(+)	1.58e-30
f_{11}	2.74e-04	1.40e-03	2.74e-04(=)	1.40e-03	2.74e-04(=)	1.40e-03	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00
f_{12}	3.16e-21	3.53e-21	5.26e-26(+)	5.38e-26	**1.57e-32**(+)	0.00e+00	**1.57e-32**(+)	0.00e+00	**1.57e-32**(+)	0.00e+00	1.21e-28(+)	1.12e-28
f_{13}	2.88e-20	2.03e-20	6.38e-25(+)	1.10e-24	**1.35e-32**(+)	2.74e-48	**1.35e-32**(+)	2.74e-48	**1.35e-32**(+)	2.74e-48	3.25e-28(+)	2.11e-28
Total number of (+/=/-):			9/2/2		9/2/2		9/2/2		9/2/2		9/2/2	
	$NP = 150$											
	$SS = 1$		$SS = 2$		$SS = 4$		$SS = 8$		$SS = 16$		$SS = NP$	
	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.
f_1	2.09e-09	1.01e-09	3.98e-12(+)	2.66e-12	2.30e-21(+)	2.11e-21	5.22e-22(+)	3.37e-22	3.44e-22(+)	2.85e-22	2.71e-29(+)	2.37e-29
f_2	7.90e-05	2.37e-05	1.71e-06(+)	5.81e-07	1.39e-10(+)	6.71e-11	6.81e-11(+)	3.69e-11	4.78e-11(+)	2.11e-11	8.49e-15(+)	4.99e-15
f_3	9.70e+00	3.48e+00	1.02e+00(+)	4.32e-01	2.41e-03(+)	1.58e-03	1.50e-03(+)	9.77e-04	1.34e-03(+)	8.51e-04	2.52e-01(+)	1.23e-01
f_4	4.73e-02	1.37e-02	7.87e-02(+)	3.47e-01	8.86e-02(+)	4.10e-01	**4.61e-05**(+)	1.49e-05	1.48e-04(+)	5.46e-04	3.60e-02(+)	3.56e-02
f_5	1.81e+01	6.96e-01	1.64e+01(+)	7.19e-01	8.14e+00(+)	1.51e+00	**7.96e+00**(+)	9.22e-01	9.01e+00(+)	1.64e+00	1.61e+01(+)	2.00e-01
f_6	**0.00e+00**	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00
f_7	1.28e-02	3.95e-03	8.98e-03(+)	2.64e-03	9.63e-03(+)	2.00e-03	5.43e-03(+)	1.56e-03	5.43e-03(+)	1.56e-03	4.79e-03(+)	4.25e-04
f_8	7.29e+03	3.04e+02	7.28e+03(=)	2.80e+02	7.16e+03(=)	2.50e+02	7.15e+03(=)	3.07e+02	7.14e+03(=)	2.54e+02	7.17e+03(=)	2.05e+02
f_9	1.86e+02	8.41e+00	1.81e+02(+)	1.18e+01	1.78e+02(+)	1.00e+01	1.67e+02(+)	1.09e+01	1.69e+02(+)	8.42e+00	1.75e+02(+)	1.00e+00
f_{10}	1.46e-05	4.38e-06	5.57e-07(+)	2.07e-07	2.98e-07(+)	1.08e-07	7.41e-12(+)	2.47e-12	4.72e-12(+)	1.78e-12	6.16e-08(+)	1.64e-08
f_{11}	2.74e-04	1.40e-03	1.91e-11(+)	2.57e-11	3.97e-12(+)	3.73e-12	**0.00e+00**(+)	0.00e+00	2.34e-11(+)	1.46e-11	1.51e-13(+)	8.50e-15
f_{12}	2.74e-10	1.88e-10	4.55e-13(+)	2.85e-13	1.01e-13(+)	7.81e-14	4.55e-23(+)	3.76e-23	3.47e-23(+)	3.39e-23	7.68e-15(+)	2.62e-15
f_{13}	1.54e-09	9.53e-10	2.28e-12(+)	1.39e-12	1.49e-21(+)	1.56e-21	1.66e-22(+)	1.69e-22	1.66e-22(+)	1.69e-22	2.04e-14(+)	1.18e-14
Total number of (+/=/-):			11/2/0		11/2/0		11/2/0		11/2/0		11/2/0	
	$PopSize = 200$											
	$SS = 1$		$SS = 2$		$SS = 4$		$SS = 8$		$SS = 16$		$SS = NP$	
	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.	Mean	St.D.
f_1	5.66e-05	2.00e-05	7.50e-07(+)	3.27e-07	5.70e-13(+)	3.13e-13	1.67e-13(+)	8.22e-14	1.15e-13(+)	7.42e-14	2.90e-14(+)	1.38e-14
f_2	1.02e-02	2.40e-03	6.63e-07(+)	7.74e-07	9.70e-07(+)	2.25e-03	9.49e-07(+)	3.00e-07	8.93e-07(+)	2.54e-07	5.90e-07(+)	1.05e-07
f_3	1.79e+02	4.18e+01	4.79e+01(+)	1.57e+01	9.86e-01(+)	2.77e-01	9.76e-01(+)	2.93e-01	8.28e-01(+)	4.68e-01	4.17e+01(+)	9.50e+00
f_4	7.45e-01	1.38e-01	2.40e-01(+)	6.31e-02	8.86e-03(+)	2.59e-03	7.34e-03(+)	2.29e-03	7.23e-03(+)	2.17e-03	1.17e-01(+)	1.80e-02
f_5	2.40e+01	5.37e-01	2.09e+01(+)	6.34e-01	1.26e+01(+)	8.92e-01	1.29e+01(+)	7.79e-01	1.28e+01(+)	8.89e-01	1.93e+01(+)	5.50e-01
f_6	**0.00e+00**	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00	**0.00e+00**(=)	0.00e+00
f_7	1.91e-02	4.75e-03	1.45e-02(+)	4.09e-03	9.10e-03(+)	2.34e-03	9.02e-03(+)	2.57e-03	9.46e-03(+)	1.92e-03	1.25e-02(+)	1.35e-03
f_8	7.39e+03	2.79e+02	7.18e+03(+)	2.59e+02	7.28e+03(+)	2.20e+02	7.19e+03(+)	4.26e+02	7.22e+03(+)	2.74e+02	7.29e+03(+)	1.00e+01
f_9	1.91e+02	1.13e+01	1.91e+02(=)	7.00e+00	1.80e+02(+)	1.07e+01	1.77e+02(+)	1.10e+01	1.80e+02(+)	9.00e+00	1.90e+02(+)	1.50e+00
f_{10}	2.34e-03	4.36e-04	2.87e-04(+)	7.05e-05	2.13e-07(+)	6.21e-08	1.47e-07(+)	4.12e-08	1.07e-07(+)	2.85e-08	9.32e-05(+)	1.90e-06
f_{11}	2.49e-04	4.18e-04	2.32e-06(+)	1.37e-06	1.53e-12(+)	1.19e-12	8.28e-13(+)	1.16e-12	5.80e-13(+)	4.31e-13	1.82e-07(+)	5.80e-08
f_{12}	1.09e-05	5.89e-06	1.05e-07(+)	5.42e-08	5.94e-14(+)	3.60e-14	1.95e-14(+)	1.13e-14	1.98e-14(+)	1.50e-14	6.66e-09(+)	3.10e-10
f_{13}	4.41e-05	2.14e-05	7.13e-07(+)	4.19e-07	3.29e-13(+)	1.84e-13	1.65e-13(+)	9.21e-14	1.14e-13(+)	8.02e-14	4.71e-08(+)	9.55e-09
Total number of (+/=/-):			11/2/0		12/1/0		12/1/0		12/1/0		12/1/0	

$SS = 16$ obtains best performance on 6 functions (f_3, f_4, f_6, f_{10}, f_{11}, f_{13}) in this testing group.

From above discussion, the STS operator promotes DE's ability of finding high accuracy solutions not only on unimodal but also on multimodal functions, and a proper SS/NP ratio lies in the range 4%-10%.

5.3 Acceleration of the Convergence Speed

To compare the convergence speed of DE/rand/1 and STS-DE/rand/1, we run them with different SS values when $NP = 100$. We select a threshold value for the objective function for each test function, which is listed in the third column of Table 3. Each algorithm is terminated when the best fitness value so far is below the predefined threshold value or the number of fitness evaluations reaches the maximum value $MAX\_FEs = 2.00E + 06$. For each test function of Group A, the algorithms were run 50 times. Here, we define the acceleration rate (AR) to show the improvement of algorithm B over algorithm A in terms of the convergence speed, as follows:

$$AR_{B/A} = \frac{Mean\_FEs_A}{Mean\_FEs_B} \tag{16}$$

Table 3 shows the results of the mean number of fitness evaluations (Mean_FEs), the success rate of converging to the predefined threshold, and the AR between STS-DE/rand/1 and DE/rand/1. The best result among the compared algorithms for each function is shown in **boldface**. As shown, STS-DE/rand/1 shows faster convergence on unimodal and multimodal functions except function f_5. f_5 is a specific function with strong relations among variables

Table 3 Results of the mean number of fes and successful rate under predefined accuracy level (threshold)

Prob.	Threshold	SS=1		SS=2			SS=4			SS=8			SS=16		
		Mean FEs	SR	Mean FEs	AR	SR	Mean FEs	AR	SR	Mean FEs	AR	SR	Mean FEs	AR	SR
f_1	1.00e-10	1.21e+05	100%	1.01e+05	1.20	100%	9.53e+04	1.27	100%	9.36e+04	1.29	100%	**9.25e+04**	1.31	100%
f_2	1.00e-10	2.09e+05	100%	1.75e+05	1.19	100%	1.65e+05	1.27	100%	1.61e+05	1.30	100%	**1.60e+05**	1.31	100%
f_3	1.00e-10	4.38e+05	100%	3.47e+05	1.26	100%	3.30e+05	1.33	100%	3.23e+05	1.36	100%	**3.18e+05**	1.38	100%
f_4	1.00e-10	NA	0%	NA	NA	0%	NA	NA	0%	NA	NA	0%	NA	NA	0%
f_5	1.00e-10	**4.63e+05**	100%	5.87e+05	0.79	100%	9.23e+05	0.50	100%	1.18e+06	0.50	100%	1.36e+06	0.34	100%
f_6	1.00e-10	3.21e+04	100%	2.70e+04	1.19	100%	2.55e+04	1.26	100%	2.47e+04	1.26	100%	**2.43e+04**	1.32	100%
f_7	1.00e-02	1.29e+05	100%	1.08e+05	1.19	100%	9.80e+04	1.32	100%	**9.71e+04**	1.32	100%	9.95e+04	1.30	100%
f_8	1.00e+03	8.14e+05	100%	6.47e+05	1.26	100%	5.80e+05	1.40	100%	**5.49e+05**	1.48	100%	5.70e+05	1.43	100%
f_9	1.00e-10	NA	0%	NA	NA	0%	NA	NA	0%	NA	NA	0%	NA	NA	0%
f_{10}	1.00e-10	1.98e+05	100%	1.63e+05	1.21	100%	1.54e+05	1.29	100%	1.51e+05	1.29	100%	**1.50e+05**	1.32	100%
f_{11}	1.00e-10	1.26e+05	100%	1.04e+05	1.21	100%	9.84e+04	1.28	98%	9.63e+04	1.28	96%	**9.56e+04**	1.32	100%
f_{12}	1.00e-10	1.13e+05	100%	9.31e+04	1.21	100%	8.73e+04	1.29	100%	8.58e+04	1.29	100%	**8.53e+04**	1.32	100%
f_{13}	1.00e-10	1.20e+05	100%	9.88e+04	1.21	100%	9.31e+04	1.29	100%	9.14e+04	1.29	100%	**9.06e+04**	1.32	100%

and a narrow valley. From Table 3, the STS-integrated algorithm maintains a high convergence speed when it has a larger subset. The main reason is that better solutions have more probability to survive into the next generation because the algorithm is under heavy selection pressure when SS is large. It is noteworthy that the algorithm is easy to fall into local optima when the selection pressure is too heavy.

5.4 Analysis of Exploration and Exploitation

An important issue when applying EAs is to balance the exploration and exploitation capabilities. In this section, we demonstrate that the proposed STS selection scheme attempts to strengthen the exploration ability at the early stage of evolution and the exploitation ability at the late stage.

To demonstrate this, we employ the rejection error index (REI), the population diversity (pd), the population fitness (pf) and the number of recently successful updates (nu) to measure the exploration and exploitation capabilities. The population diversity in the *G*th generation is calculated as follows:

$$pd^G = \sum_{i=1}^{NP-1} \sum_{j=i+1}^{NP} D(X_i^G, X_j^G) \tag{17}$$

where $D(X_i^G, X_j^G)$ is the Euclidean distance between the *i*th vector and the *j*th vector at generation G.

Similarly, the population fitness in the *G*th generation is calculated by:

$$pf^G = \sum_{i=1}^{NP} f(X_i^G). \tag{18}$$

In most cases, an algorithm enhances the search ability if it can constantly generate successful solutions. This phenomenon can be shown by evaluating the number of recently successful updates nu^G in the *G*th generation, which is calculated as follows:

$$nu^G = \sum_{i=1}^{NP} q_i^G, \tag{19}$$

where

$$q_i^G = \begin{cases} 1, & if\ X_i^G\ is\ updated \\ 0, & otherwise \end{cases} \tag{20}$$

If the parameter nu^G is in a high level, it means that the algorithm maintains a high update possibility and has the ability of generating successful solutions.

Obviously, an algorithm tends to successfully explore the search space if it has a high level of population diversity and a high possibility of successful updates. Furthermore, it tends to well exploit if it has a low level of population fitness and maintains a high amount of successful updates.

Fig. 3 presents the accumulation of REI, pd, pf and nu of running DE/rand/1 with the STS operator on the 30-dimensional multimodal functions (f_8-f_{13}) in the testing group A. As shown in Fig. 3, the value of SS has a significant impact on REI value, where a big value of $SS > 1$ reduces REI. More specifically, compared with the original one-to-one survivor selection operator, which is the special case of the STS selection with $SS = 1$, the proposed STS method with $SS > 1$ does not seriously reduce the population diversity. This means that the DE with the STS operator integrated still has the exploratory capacity competitive to the DE without the STS operator. In addition, it is noticeable that STS-DE/rand/1 with $SS > 1$ maintains a higher number of successful updates (nu) than DE/rand/1. The average fitness is decreased by using the STS operator with $SS > 1$, which demonstrates that the DE algorithm with the STS operator has a strong ability of finding potential solutions and balancing between exploration and exploitation.

5.5 Comparison of Five Survival Selection Schemes

In this section, the STS selection scheme is compared with the other four survival selection methods, namely, the worst survival selection (CW), which is outstanding in the convergence speed, the closest survival selection (CC) Tagawa (2009), which is excellent in the robustness, DCN-THR-REF Segura et al. (2013) and tournament selection (TS)

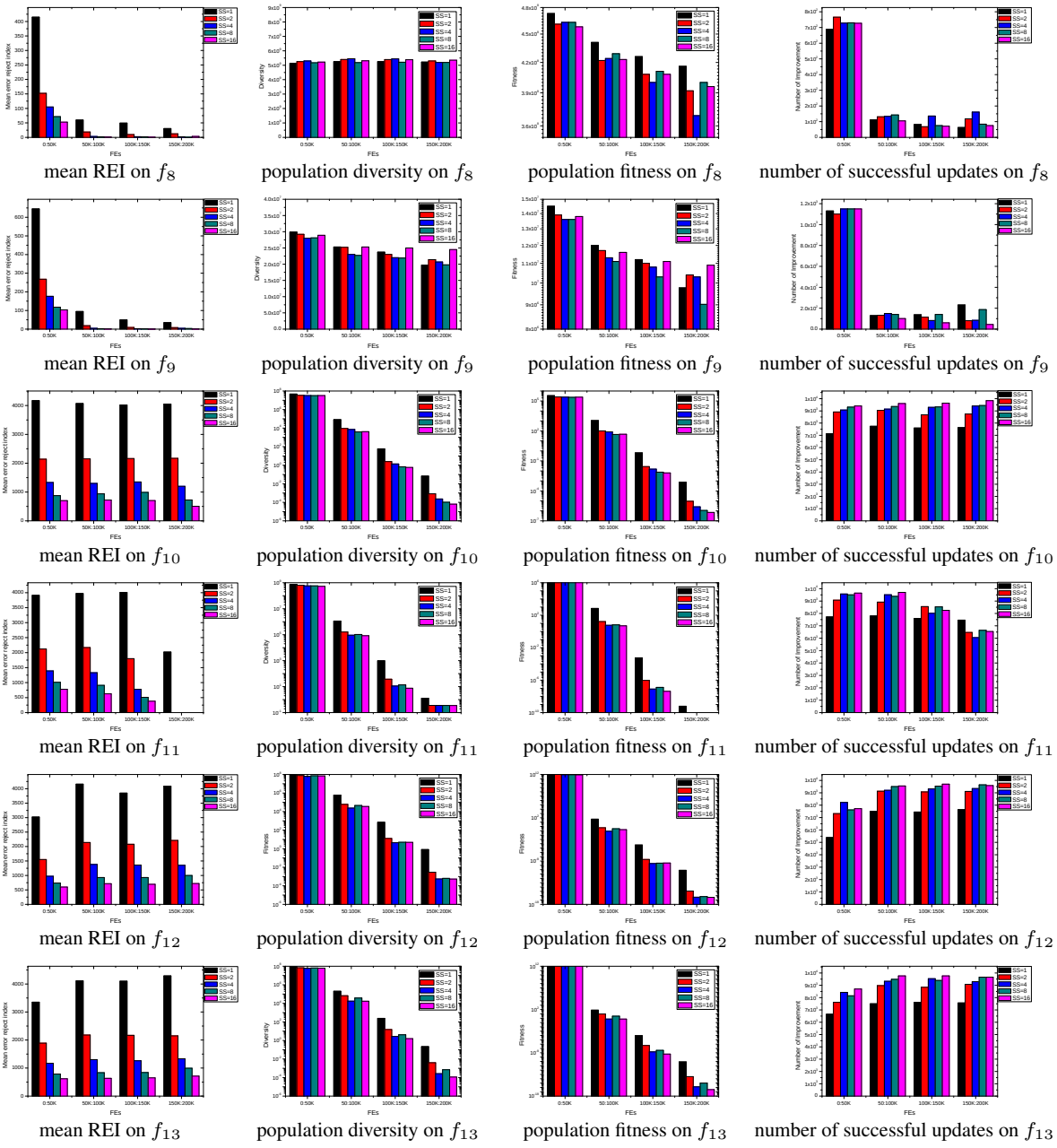

Fig. 3 The accumulation of REI, population diversity, population fitness and number of successful updates of running DE/rand/1/bin on the 30-dimensional multimodal functions of Group A, where $NP = 100$ with $SS = 1$, $SS = 2$, $SS = 4$, $SS = 8$ and $SS = 16$, respectively.

(Abbas et al., 2015). Again, we integrate the five selection methods into the DE/rand/1 mutation strategy, denoted STS_DE, CW_DE, CC_DE, DCN-THR-REF_DE and T-S_DE, respectively. The scaling factor F and crossover rate CR are set to 0.5 and 0.9, respectively. Depending on the analysis in Section 5.2, the subset size SS is set to $4\% \times NP$ for the STS selection scheme. Each algorithm was run 30 independent times for every function. The results are reported in Table 4.

A close inspection in Table 4 indicates that the performance of STS_DE is clearly and constantly superior to the other four survival selections in most of the unimodal and multimodal functions.The proposed STD_DE has 11 best average fitness performances for unimodal functions f_1- f_7 and multimodal functions f_9, f_{10}, f_{12}, f_{13}. CW_DE has the second best average fitness performance for unimodal function f_6 and multimodal functions f_8, f_{12}, f_{13}. Compared with STS_DE, CW_DE performs worse on 7 functions, similarly on 5 functions and only outperforms on one function f_8. CC_DE does not obtain the best average fitness on any testing functions and exhibits worse on all functions in Wilcoxon's test with STS_DE, so it performs

Table 4 The error values for different survival selection schemes at $NP = 100$, $SS = 4$, $D = 30$, after 200,000 fitness evaluations

	STS_DE	CW_DE	CC_DE	DCN-THR-REFS_DE	TS_DE
	Mean St.D.	Mean St.D.	Mean St.D.	Mean St.D.	Mean St.D.
f_1	**5.43e-42** 4.62e-42	4.31e-34(−) 5.91e-34	2.30e-01(−) 4.09e-02	1.03e+04(−) 3.17e+03	8.06e-28(−) 1.05e-27
f_2	**1.16e-20** 1.05e-20	6.89e-17(−) 4.05e-17	3.96e+00(−) 2.10e+00	2.41e-07(−) 4.80e-07	5.47e-14(−) 3.52e-14
f_3	**4.47e-09** 3.83e-09	5.86e-06(−) 6.70e-06	5.19e+03(−) 1.06e+03	1.37e+04(−) 5.13e+03	2.80e-05(−) 4.07e-05
f_4	**5.65e-01** 1.06e+00	1.14e+00(−) 1.52e+00	3.40e+01(−) 2.79e+00	1.02e+00(−) 1.42e+00	7.59e-01(=) 7.35e-01
f_5	**1.53e+01** 2.50e+00	1.75e+01(−) 1.58e+00	5.72e+02(−) 1.01e+02	1.08e+07(−) 5.15e+06	2.13e+01(−) 1.01e+00
f_6	**0.00e+00** 0.00e+00	**0.00e+00**(=) 0.00e+00	3.33e-02(−) 1.80e-01	9.05e+03(−) 2.88e+03	**0.00e+00**(=) 0.00e+00
f_7	**2.96e-03** 7.85e-04	4.65e-03(−) 1.28e-03	4.76e-02(−) 1.22e-02	5.21e-03(−) 1.68e-03	4.84e-03(−) 1.56e-03
f_8	6.66e+03 6.45e+02	**6.60e+03**(+) 6.27e+02	7.42e+03(−) 2.60e+02	6.80e+03(−) 1.87e+02	6.74e+03(−) 5.09e+02
f_9	**1.02e+02** 2.53e+01	1.41e+02(−) 2.63e+01	2.16e+02(−) 9.97e+00	1.52e+02(−) 1.13e+01	1.43e+02(−) 2.20e+01
f_{10}	**4.00e-15** 1.58e-30	4.95e-15(=) 1.57e-15	2.21e+00(−) 1.83e-01	3.05e-13(−) 1.21e-13	8.26e-15(=) 2.14e-15
f_{11}	2.74e-04 1.40e-03	2.47e-04(=) 1.33e-03	7.96e-01(−) 5.94e-02	**0.00e+00**(=) 0.00e+00	**0.00e+00**(=) 0.00e+00
f_{12}	**1.57e-32** 0.00e+00	**1.57e-32**(=) 2.74e-48	2.28e+01(−) 5.68e+00	1.46e+07(−) 9.49e+06	2.86e-29(−) 2.86e-29
f_{13}	**1.35e-32** 2.74e-48	**1.35e-32**(=) 5.47e-48	8.44e+01(−) 9.53e+01	6.52e+07(−) 1.57e+07	4.79e-28(−) 6.21e-28
Total number of (+/=/-):		1/5/7	0/0/13	0/1/12	0/4/9

worst in this survival selection test. In the comparison with STS_DE, DCN-THR-REF_DE exhibits similar performance on one function f_{11}. TS_DE, which uses the common selection method in genetic algorithm, exhibits similar performance on 4 functions, worse performance on other 9 functions. From above discussion, STS_DE is a reliable selection scheme although it greedily selects the better individuals from the target subset and the corresponding trail subset.

5.6 Enhancing the Performance of DE Algorithms

To further verify the performance of the STS operator, it is integrated into six state-of-the-art DE algorithms, which are listed below:

1. ODE Rahnamayan et al. (2008) with $NP = 100$, $F = 0.5$, $Cr = 0.9$, $Jr = 0.3$;
2. JADE Zhang and Sanderson (2009) with an external archive, $NP = 100$, $c = 0.1$, $p = 0.05$;
3. DEGL/SAW Das et al. (2009) with $NP = 10 \times D$, $F = 0.8$, $Cr = 0.9$;
4. EPSDE Mallipeddi et al. (2011) with $NP = 50$;
5. FiADE Ghosh et al. (2011) with $NP = 50$;
6. MGBDE Wang et al. (2013) with $NP = 100$.

To make a fair comparison, all parameters of DE variants above were kept the same as used in their original literature. According to the discussion of Section A and the computational cost, the subset size SS is set to $4\% \times NP$. For the 2014 IEEE CEC benchmark functions, the maximum number of function evaluations was set to $10000 \times D$. Each algorithm was executed independently 30 times to obtain the mean solution error and standard deviation. Similarly, the Wilcoxon's rank-sum test at the 0.05 significance level is employed to judge the performance difference between the original DE algorithms and their corresponding STS variants. The experimental results regarding the error values are given in Tables 5, 6 and 7. Figs. 4, 5, 6 and 7 show the convergence curves of the six DE algorithms and their corresponding STS variants on Group B functions cf_1, cf_5, cf_{17} and cf_{30}, respectively.

For the opposition-based DE, one can observe that STS-ODE outperforms ODE in 7 cases (cf_2, cf_3, cf_6, cf_{22}, and $cf_{28} - cf_{30}$) and exhibits similar performance in 21 functions. JADE is substantially enhanced by the STS operator. STS-JADE exhibits significantly better or similar performance in 26 out of the 30 functions. More specifically, in the unimodal functions $cf_1 - cf_3$, STS-JADE exhibits similar performance to JADE. In the simple mutilmodal and hybrid functions $cf_4 - cf_{22}$, STS-JADE exhibits either better or similar performance except for cf_4 and cf_{22}. For the composition functions, STS-JADE outperforms JADE in three cases (cf_{27}, cf_{29}, cf_{30}) and exhibits similar performance in cf_{23}, cf_{26} and cf_{28}. STS-DEGL/SAW either outperforms DEGL/SAW or performs equally well. In more detail, DEGL /SAW is improved by the STS operator in 13 functions, while the performance of the two algorithms is not statistically different in 17 functions (i.e., cf_5, cf_7, $cf_9 - cf_{15}$, cf_{18}, cf_{20} and $cf_{23} - cf_{28}$). STS-EPSDE outperforms EPSDE in 14 functions. Specifically, STS-EPSDE achieves significantly better performance on two unimodal functions (cf_1 and cf_3) and eight multimodal functions ($cf_5 - cf_6$ and $cf_8 - cf_{13}$). EPSDE outperforms STS-EPSDE in 8 functions, cf_4, cf_7, $cf_{17} - cf_{18}$, $cf_{20} - cf_{22}$ and cf_{24}, most of which are hybrid functions. Similarly, STS-FiADE shows better performance in 17 functions and similar performance in 6 functions. STS-MGBDE demonstrates either similar or significantly better performance in 19 functions, among which STS-MGBDE attains a statistically significant performance improvement for 11 functions.

In order to further compare the performance of all considered algorithms on the 2014 IEEE CEC benchmark functions, we employ the Friedman's test as done in García et al. (2009, 2010). Table 8 shows the average rankings of the six

Table 5 The error values of ODE, STS-ODE, JADE and STS-JADE on the 30-dimensional 2014 IEEE CEC benchmark functions

	ODE		STS-ODE			JADE		STS-JADE		
	Mean	St.D.	Mean	St.D.		Mean	St.D.	Mean	St.D.	
cf_1	1.05e+05	5.93e+04	1.05e+05	8.77e+04	$=$	1.27e+04	1.02e+04	1.45e+04	1.30e+04	$=$
cf_2	4.78e-13	6.85e-13	1.60e-14	2.14e-14	$+$	0.00e+00	0.00e+00	9.87e-24	2.72e-23	$=$
cf_3	2.21e-16	2.03e-16	2.04e-17	2.38e-17	$+$	8.49e+01	7.00e+01	2.30e+02	3.86e+02	$=$
cf_4	4.86e-01	2.63e-01	2.96e+00	1.21e+01	$-$	3.82e-07	1.64e-06	2.17e-01	2.45e-01	$-$
cf_5	2.06e+01	1.35e-01	2.06e+01	1.56e-01	$=$	2.03e+01	3.00e-02	2.01e+01	4.82e-02	$+$
cf_6	2.04e-01	8.56e-01	5.59e-01	2.46e+00	$+$	1.15e+01	1.08e+00	3.16e+00	2.48e+00	$+$
cf_7	1.78e-16	1.69e-16	2.18e-16	1.71e-16	$=$	4.89e-07	2.57e-06	3.45e-03	5.87e-03	$=$
cf_8	1.56e+02	1.46e+01	1.59e+02	1.80e+01	$=$	2.81e-13	1.42e-13	0.00e+00	0.00e+00	$+$
cf_9	1.87e+02	7.98e+00	1.85e+02	9.33e+00	$=$	3.11e+01	3.98e+00	2.69e+01	6.10e+00	$+$
cf_{10}	2.85e+03	6.12e+02	3.49e+03	8.76e+02	$-$	1.95e-01	5.55e-02	2.96e-01	3.10e-01	$=$
cf_{11}	4.21e+03	1.25e+03	4.26e+03	1.16e+03	$=$	1.69e+03	2.41e+02	1.60e+03	4.01e+02	$+$
cf_{12}	6.94e-01	3.22e-01	7.93e-01	3.24e-01	$=$	2.81e-01	2.86e-02	1.48e-01	4.40e-02	$+$
cf_{13}	3.57e-01	4.53e-02	3.66e-01	3.97e-02	$=$	2.35e-01	3.38e-02	1.62e-01	3.85e-02	$+$
cf_{14}	2.74e-01	3.54e-02	2.70e-01	3.67e-02	$=$	2.26e-01	2.83e-02	2.42e-01	4.19e-02	$=$
cf_{15}	1.62e+01	1.07e+00	1.59e+01	8.80e-01	$=$	3.83e+00	4.93e-01	3.28e+00	9.38e-01	$+$
cf_{16}	1.17e+01	6.48e-01	1.16e+01	8.71e-01	$=$	9.49e+00	2.81e-01	8.61e+00	6.50e-01	$+$
cf_{17}	1.52e+03	1.75e+02	1.53e+03	1.45e+02	$=$	2.39e+05	3.70e+05	6.84e+04	1.14e+05	$+$
cf_{18}	5.68e+01	4.90e+00	5.61e+01	5.85e+00	$=$	3.60e+03	5.57e+03	4.12e+02	1.02e+03	$+$
cf_{19}	4.63e+00	3.29e-01	4.65e+00	2.83e-01	$=$	4.94e+00	8.42e-01	3.89e+00	7.17e-01	$+$
cf_{20}	3.84e+01	4.44e+00	3.70e+01	3.93e+00	$=$	4.98e+03	2.52e+03	5.15e+03	4.78e+03	$=$
cf_{21}	8.14e+02	1.08e+02	8.32e+02	1.22e+02	$=$	1.36e+05	8.88e+04	7.79e+04	9.68e+04	$+$
cf_{22}	8.98e+01	8.49e+01	5.94e+01	5.60e+01	$+$	1.92e+02	6.24e+01	2.85e+02	1.32e+02	$-$
cf_{23}	3.15e+02	0.00e+00	3.15e+02	0.00e+00	$=$	3.15e+02	0.00e+00	3.15e+02	0.00e+00	$=$
cf_{24}	2.17e+02	9.27e+00	2.17e+02	9.55e+00	$=$	2.25e+02	1.78e+00	2.27e+02	3.13e+00	$-$
cf_{25}	2.00e+02	0.00e+00	2.00e+02	7.48e-01	$=$	2.07e+02	2.42e+00	2.10e+02	2.45e+00	$-$
cf_{26}	1.00e+02	0.00e+00	1.07e+02	2.49e+01	$=$	1.00e+02	0.00e+00	1.00e+02	0.00e+00	$=$
cf_{27}	3.54e+02	8.49e+01	3.81e+02	6.03e+01	$=$	3.72e+02	4.68e+01	3.47e+02	3.91e+01	$+$
cf_{28}	6.35e+02	2.63e+02	5.46e+02	2.84e+02	$+$	8.68e+02	2.74e+01	8.56e+02	4.62e+01	$=$
cf_{29}	4.13e+02	2.32e+02	3.09e+02	2.23e+02	$+$	1.34e+03	4.88e+02	9.12e+02	3.45e+02	$+$
cf_{30}	7.51e+02	1.91e+02	6.96e+02	1.69e+02	$+$	2.10e+03	5.80e+02	1.74e+03	4.27e+02	$+$
Total number of $(+/=/-)$: 7/21/2						Total number of $(+/=/-)$: 16/10/4				

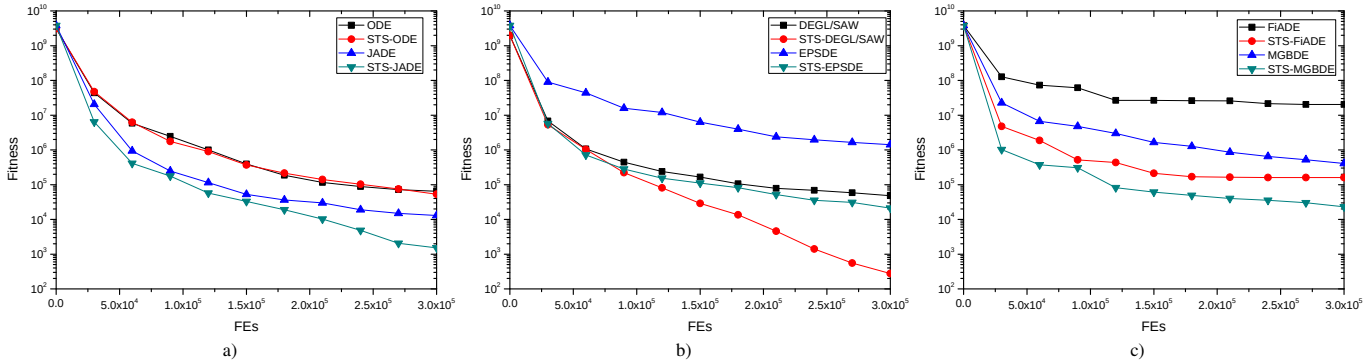

Fig. 4 The convergence curves of the six DE algorithms and their corresponding STS variants on 2014 CEC benchmark functions of unimodal function cf_1.a)The convergence curves of ODE, STS-ODE, JADE and STS-JADE on cf_1. b)The convergence curves of DEGL/SAW, STS-DEGL/SAW,EPSDE, STS-EPSDE on cf_1. c)The convergence curves of FiADE, STS-FiADE, MGBDE and STS-MGBDE on cf_1

state-of-the-art DE variants and their STS versions over the 30-dimensional 2014 IEEE CEC benchmark functions. The best ranking is shown in **bold face**. As seen, the best average ranking is obtained by the STS-L-SHADE and L-SHADE. Furthermore, all STS versions outperform the non-STS versions except L-SHADE.

6 Conclusions

In this paper, an effective and efficient STS survival selection operator is proposed to enhance the performance of DE algorithms. The STS operator aims to appropriately accept trial vectors with better objective function values and reject trial vectors with worse objective function values as many as possible. This selection scheme divides the target and trial populations into a number of subsets and then selects the

Table 6 The error values of DEGL/SAW, STS-DEGL/SAW, EPSDE and STS-EPSDE on the 30-dimensional 2014 IEEE CEC benchmark functions

	DEGL/SAW		STS-DEGL/SAW			EPSDE		STS-EPSDE		
	Mean	St.D.	Mean	St.D.		Mean	St.D.	Mean	St.D.	
cf_1	1.33e+04	1.94e+04	3.48e+03	4.89e+03	+	1.37e+06	6.37e+05	4.48e+04	2.93e+04	+
cf_2	1.32e-05	6.80e-05	1.50e-11	7.01e-11	+	0.00e+00	0.00e+00	1.11e-22	2.77e-22	=
cf_3	3.30e-09	1.77e-08	1.86e-14	9.27e-14	+	7.89e-20	7.17e-20	5.06e-24	1.16e-23	+
cf_4	2.11e+00	1.14e+01	5.23e-05	2.10e-04	+	8.05e-02	7.14e-02	3.10e-01	2.10e-01	−
cf_5	2.09e+01	5.96e-02	2.09e+01	5.96e-02	=	2.07e+01	5.59e-02	2.05e+01	8.62e-02	+
cf_6	3.08e-02	1.25e-01	6.28e-02	2.35e-01	+	2.09e+01	1.43e+00	2.39e+00	2.48e+00	+
cf_7	2.38e-03	5.15e-03	3.04e-03	5.52e-03	=	0.00e+00	0.00e+00	2.30e-03	4.26e-03	−
cf_8	1.65e+02	9.59e+00	1.59e+02	9.18e+00	+	3.01e+01	2.69e+00	1.60e+01	4.10e+00	+
cf_9	1.78e+02	1.22e+01	1.75e+02	1.13e+01	=	1.26e+02	7.64e+00	4.84e+01	1.35e+01	+
cf_{10}	6.06e+03	2.48e+02	6.06e+03	2.70e+02	=	1.01e+03	1.47e+02	2.42e+02	1.32e+02	+
cf_{11}	6.74e+03	3.19e+02	6.75e+03	2.89e+02	=	4.87e+03	3.35e+02	2.77e+03	5.53e+02	+
cf_{12}	2.47e+00	2.58e-01	2.39e+00	3.10e-01	=	1.03e+00	1.46e-01	3.10e-01	1.38e-01	+
cf_{13}	2.81e-01	6.16e-02	2.94e-01	4.24e-02	=	3.45e-01	3.00e-02	1.76e-01	4.85e-02	+
cf_{14}	2.53e-01	3.65e-02	2.50e-01	4.03e-02	=	2.57e-01	3.17e-02	2.43e-01	3.47e-02	=
cf_{15}	1.52e+01	8.83e-01	1.48e+01	9.58e-01	=	1.23e+01	1.08e+00	4.18e+00	1.25e+00	=
cf_{16}	1.24e+01	2.56e-01	1.22e+01	2.11e-01	+	1.15e+01	2.48e-01	1.02e+01	7.71e-01	=
cf_{17}	1.18e+03	4.00e+02	1.07e+03	2.27e+02	+	2.97e+03	1.16e+03	9.19e+03	1.07e+04	−
cf_{18}	5.84e+01	1.23e+01	5.26e+01	6.86e+00	=	6.79e+01	6.89e+00	9.45e+02	3.29e+03	−
cf_{19}	4.61e+00	5.35e-01	4.32e+00	5.55e-01	+	5.30e+00	8.59e-01	3.37e+00	7.01e-01	+
cf_{20}	3.53e+01	5.88e+00	3.38e+01	4.96e+00	=	3.38e+01	5.02e+00	3.80e+02	8.01e+02	−
cf_{21}	7.61e+02	1.88e+02	6.39e+02	1.67e+02	+	8.65e+02	1.57e+02	7.72e+03	1.42e+04	−
cf_{22}	2.49e+02	1.17e+02	1.88e+02	1.41e+02	+	1.32e+02	7.38e+01	4.59e+02	1.90e+02	−
cf_{23}	3.15e+02	0.00e+00	3.15e+02	0.00e+00	=	3.15e+02	0.00e+00	3.15e+02	0.00e+00	=
cf_{24}	2.20e+02	7.87e+00	2.21e+02	7.09e+00	=	2.23e+02	9.52e-01	2.25e+02	3.22e+00	−
cf_{25}	2.03e+02	0.00e+00	2.03e+02	0.00e+00	=	2.05e+02	7.72e-01	2.04e+02	8.46e-01	+
cf_{26}	1.00e+02	0.00e+00	1.00e+02	0.00e+00	=	1.00e+02	0.00e+00	1.03e+02	1.80e+01	=
cf_{27}	3.63e+02	4.62e+01	3.50e+02	5.00e+01	=	3.69e+02	4.86e+01	3.68e+02	5.07e+01	=
cf_{28}	7.98e+02	6.95e+01	8.06e+02	7.74e+01	=	9.70e+02	2.84e+01	8.79e+02	3.88e+01	+
cf_{29}	3.14e+02	2.56e+02	2.54e+02	2.33e+02	+	1.03e+03	1.54e+02	5.56e+02	2.06e+02	+
cf_{30}	6.04e+02	2.42e+02	4.30e+02	8.45e+01	+	1.30e+03	1.89e+02	1.49e+03	6.37e+02	=
Total number of $(+/=/-)$: 13/17/0						Total number of $(+/=/-)$: 14/8/8				

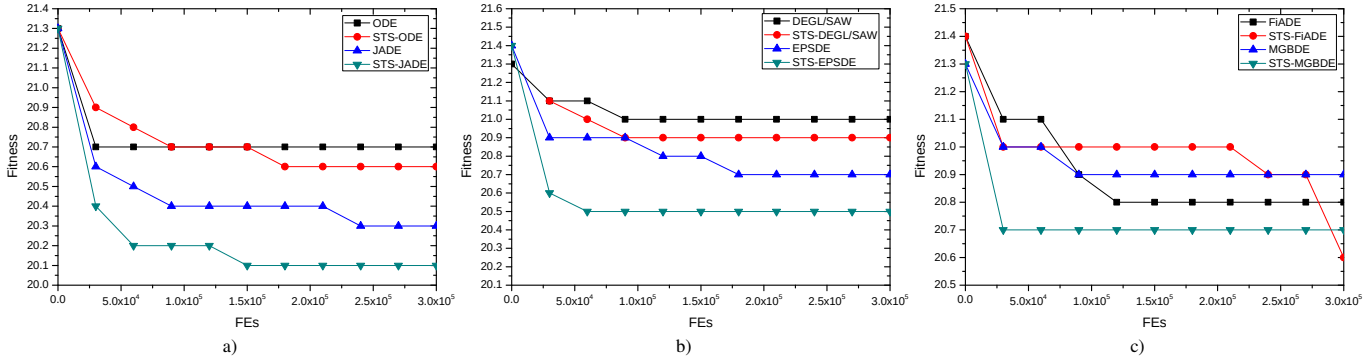

Fig. 5 The convergence curves of the six DE algorithms and their corresponding STS variants on 2014 CEC benchmark functions of unimodal function cf_5.a)The convergence curves of ODE, STS-ODE, JADE and STS-JADE on cf_5. b)The convergence curves of DEGL/SAW, STS-DEGL/SAW,EPSDE, STS-EPSDE on cf_5. c)The convergence curves of FiADE, STS-FiADE, MGBDE and STS-MGBDE on cf_5

best vectors from the corresponding subsets to survive into the next generation.

First of all, we conducted the research on the relationship between the population size and the subset size and the analysis of exploration and exploitation. Then, we compared the STS operator with the other four survival selection schemes. The results show that the proposed approach is a reliable selection scheme although it greedily selects the better individuals from the target subset and the corresponding trail subset. At last, we incorporated this STS selection scheme with six state-of-the-art DE variants. The experimental results show that the proposed STS operator improves the performance of considered DE algorithms in terms of the solution error measure. To conclude, the proposed STS selection operator accelerates the convergence of DE algorithms by keeping more promising solutions.

Table 7 The error values of FiADE, STS-FiADE, MGBDE and STS-MGBDE on the 30-dimensional 2014 IEEE CEC benchmark functions

	FiADE		STS-FiADE			MGBDE		STS-MGBDE		
	Mean	St.D.	Mean	St.D.		Mean	St.D.	Mean	St.D.	
cf_1	2.03e+07	4.65e+06	4.31e+06	1.27e+07	+	3.85e+06	2.08e+06	2.22e+04	2.28e+04	+
cf_2	1.95e-03	1.03e-03	4.52e+04	1.71e+05	=	2.45e-17	9.35e-17	1.10e-21	1.63e-21	+
cf_3	5.32e+00	4.35e+00	3.55e+03	5.41e+03	−	3.36e-09	1.10e-08	3.10e-17	2.19e-16	+
cf_4	1.26e+02	1.39e+01	2.21e+01	3.82e+01	+	5.19e+01	3.35e+01	2.03e+01	3.28e+01	+
cf_5	2.09e+01	5.42e-02	2.08e+01	1.45e-01	+	2.09e+01	4.46e-02	2.07e+01	1.57e-01	+
cf_6	2.09e+01	1.22e+00	7.44e+00	4.49e+00	+	3.17e+00	2.35e+00	6.87e+00	2.66e+00	−
cf_7	1.73e-02	7.54e-02	5.75e-04	2.18e-03	+	5.12e-03	7.47e-03	9.51e-03	8.26e-03	−
cf_8	1.81e-05	9.82e-06	2.45e+01	5.79e+00	−	4.70e+00	2.34e+00	3.93e+01	1.14e+01	−
cf_9	1.12e+02	1.03e+01	9.60e+01	2.05e+01	+	7.10e+01	3.84e+01	7.06e+01	2.31e+01	+
cf_{10}	3.97e+00	1.52e+00	2.98e+02	1.27e+02	−	3.10e+01	4.85e+01	6.10e+02	2.92e+02	−
cf_{11}	4.08e+03	1.89e+02	4.40e+03	9.86e+02	=	4.73e+03	1.27e+03	3.86e+03	9.75e+02	+
cf_{12}	1.01e+00	1.07e-01	4.41e-01	1.43e-01	+	1.87e+00	1.95e-01	6.94e-01	4.04e-01	+
cf_{13}	4.85e-01	4.79e-02	3.44e-01	5.81e-02	+	3.09e-01	4.35e-02	3.64e-01	9.93e-02	=
cf_{14}	2.69e-01	2.99e-02	3.23e-01	8.22e-02	−	2.61e-01	8.70e-02	3.36e-01	1.06e-01	−
cf_{15}	1.33e+01	8.72e-01	9.54e+00	3.70e+00	+	1.25e+01	2.59e+00	7.76e+00	3.38e+00	+
cf_{16}	1.23e+01	2.44e-01	1.02e+01	1.09e+00	+	1.17e+01	3.00e-01	1.13e+01	7.54e-01	+
cf_{17}	7.21e+05	2.13e+05	2.09e+05	5.16e+05	+	2.01e+05	1.64e+05	7.46e+03	5.23e+03	+
cf_{18}	1.20e+04	5.30e+03	2.30e+03	3.86e+03	+	3.28e+03	3.46e+03	2.59e+03	3.11e+03	+
cf_{19}	1.01e+01	7.88e-01	8.32e+00	1.60e+01	+	5.60e+00	1.21e+00	4.65e+00	1.12e+00	+
cf_{20}	5.83e+02	2.40e+02	1.04e+04	7.13e+03	−	7.90e+01	1.43e+01	5.26e+02	8.72e+02	−
cf_{21}	8.29e+04	3.29e+04	8.52e+04	9.21e+04	=	1.38e+04	1.17e+04	5.16e+03	1.82e+04	+
cf_{22}	2.40e+02	6.42e+01	5.88e+02	1.70e+02	−	1.90e+02	9.32e+01	3.83e+02	1.77e+02	−
cf_{23}	3.15e+02	0.00e+00	3.15e+02	1.80e-01	=	3.15e+02	0.00e+00	3.15e+02	0.00e+00	=
cf_{24}	2.25e+02	6.80e-01	2.26e+02	3.57e+00	=	2.26e+02	4.11e+00	2.32e+02	6.82e+00	=
cf_{25}	2.10e+02	9.52e-01	2.05e+02	1.83e+00	+	2.05e+02	1.38e+00	2.07e+02	2.82e+00	−
cf_{26}	1.00e+02	4.90e-01	1.00e+02	1.80e-01	=	1.08e+02	2.69e+01	1.08e+02	2.69e+01	=
cf_{27}	4.40e+02	1.22e+01	4.24e+02	4.80e+01	+	3.90e+02	5.02e+01	4.68e+02	8.09e+01	−
cf_{28}	1.04e+03	4.61e+01	9.41e+02	5.95e+01	+	8.16e+02	4.95e+01	9.46e+02	9.99e+01	−
cf_{29}	2.02e+03	2.48e+02	1.17e+03	4.06e+02	+	1.11e+03	3.09e+02	6.34e+02	2.17e+02	+
cf_{30}	3.60e+03	9.45e+02	4.10e+03	3.44e+03	=	1.48e+03	5.30e+02	2.19e+03	8.04e+02	−
Total number of $(+/=/-)$: 17/7/6						Total number of $(+/=/-)$: 15/4/11				

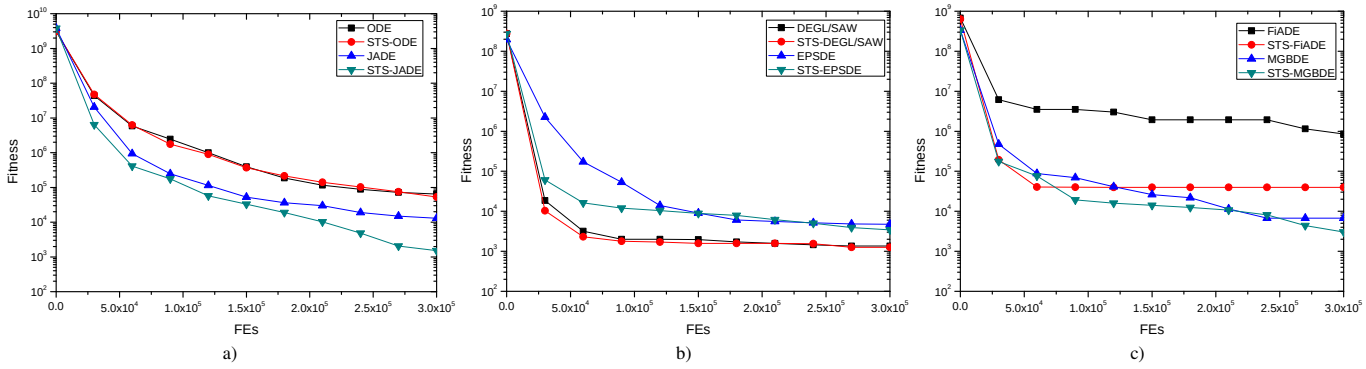

Fig. 6 The convergence curves of the six DE algorithms and their corresponding STS variants on 2014 CEC benchmark functions of unimodal function cf_{17}.a)The convergence curves of ODE, STS-ODE, JADE and STS-JADE on cf_{17}. b)The convergence curves of DEGL/SAW, STS-DEGL/SAW,EPSDE, STS-EPSDE on cf_{17}. c)The convergence curves of FiADE, STS-FiADE, MGBDE and STS-MGBDE on cf_{17}

One possible future work includes the application of the proposed work to the other population-based EAs such as particle swarm optimization and more classic/complex multimodal low dimension problems. In addition, it is noteworthy that the STS operator does not improve the performance of L-SHADE significantly, which also remains a challenge for future work.

Acknowledgment

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant EP/K001310/1 and the self-determined research funds of CCNU from the colleges' basic research and operation of MOE (No. CCNU15A05063).

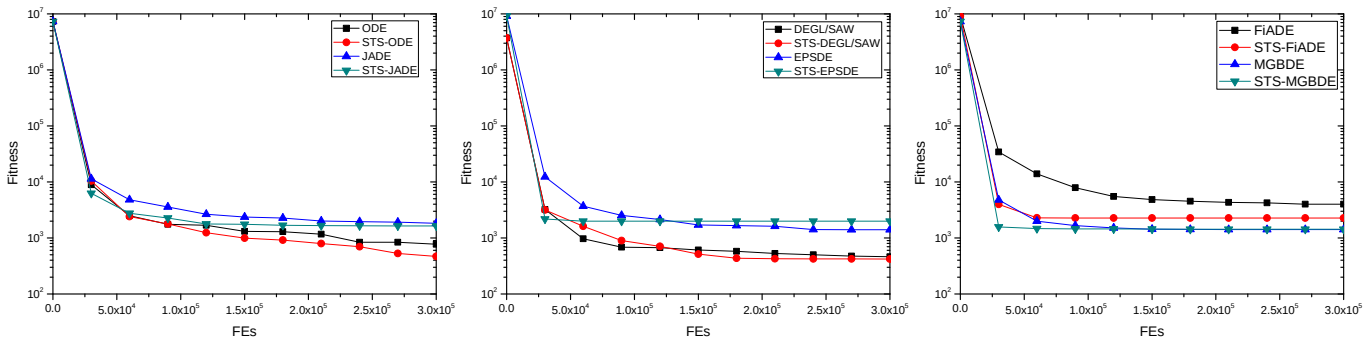

Fig. 7 The convergence curves of the six DE algorithms and their corresponding STS variants on 2014 CEC benchmark functions of unimodal function cf_{30}.a)The convergence curves of ODE, STS-ODE, JADE and STS-JADE on cf_{30}. b)The convergence curves of DEGL/SAW, STS-DEGL/SAW,EPSDE, STS-EPSDE on cf_{30}. c)The convergence curves of FiADE, STS-FiADE, MGBDE and STS-MGBDE on cf_{30}

Table 8 Average rankings achieved by the Friedman test on the 30-dimensional 2014 IEEE CEC benchmark functions

Algorithms	Average Rankings
STS-JADE	3.46
STS-DEGL/SAW	4.3
STS-EPSDE	4.3
JADE	4.53
STS-ODE	4.60
ODE	4.66
DEGL/SAW	4.7
EPSDE	5.16
STS-MGBDE	5.63
MGBDE	6.00
STS-FiADE	6.66
FiADE	7.53

Compliance with Ethical Standards

Conflict of Interest: Jinglei Guo declares that she has no conflict of interest. Zhijian Li declares that he has no conflict of interest. Shengxiang Yang declares that he has no conflict of interest.

Ethical approval: This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abbas Q, Ahmad J, Jabeen H (2015). A novel tournament selection based differential evolution variant for continuous optimization problems. Mathematical Problems in Engineering 2015

Brest J, Greiner S, Bošković B, Mernik M, Zumer V (2006). Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans. Evol. Comput. 10(6): 646–657

Brest J, Maučec M S (2009). Population size reduction for the differential evolution algorithm. Applied Intelligence 29(3): 228–247

Brest J, Maučec MS and Bošković B (2017). Single objective real-parameter optimization: algorithm jSO. In Proceedings of 2017 IEEE Congr. Evol. Comput. pp. 1311–1318

Cai H, Chung C, Wong K (2008). Application of differential evolution algorithm for transient stability constrained optimal power flow. IEEE Trans. Power Syst. 23(2): 719–728

Cruz IL, Van WLG, Van SG (2003). Efficient differential evolution algorithms for multimodal optimal control problems. Appl. Soft Comput. 3(2): 97–122

Das S, Abraham A, Chakraborty UK, Konar A (2009). Differential evolution using a neighborhood-based mutation operator. IEEE Trans. Evol. Comput. 13(3): 526–553

Das S, Suganthan PN (2011). Differential evolution: A survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15(1): 4–31

Das S, Mullick SS, Suganthan PN (2016). Recent advances in differential evolution: An updated survey. Swarm and Evolutionary Computation 27: 1–30

Epitropakis MG, Tasoulis DK, Pavlidis NG, Vrahatis MN (2011). Enhancing differential evolution utilizing proximity-based mutation operators. IEEE Trans. Evol. Comput. 15(1): 99–119

García S, Molina D, Lozano M, Herrera F (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. Journal of Heuristics 15(6): 617–644

García S, Fernández A, Luengo J, Herrera F (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Inform. Sci. 180(10): 2044–2064

Ghosh A, Das S, Chowdhury A, Giri R(2011). An improved differential evolution algorithm with fitness-based adaptation of the control parameters. Inform. Sci. 181(18): 3749–3765

Gong W, Cai Z (2013). Differential evolution with ranking-based mutation operators. IEEE Trans. Cybern. 43(6): 2066–2081

Guo SM, Yang CC, Hsu PH, Tsai JC (2014). Improving differential evolution with successful-parent-selecting framework. IEEE Trans. Evol. Comput. 19(5): 717–730

Li YL, Zhan ZH, Gong YJ, Chen WN, Zhang J, Li Y (2015). Differential evolution with an evolution path: A deep evolutionary algorithm. IEEE Trans. Cybern. 45(9): 1798–1810

Liang JJ, Qu BY, Suganthan PN (2014). Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, 2013

Liu ZZ, Wang Y, Yang S, et al (2016). Differential evolution with a two-stage optimization mechanism for numerical optimization, In: Proc. 2013 IEEE Congr. Evol. Comput. pp. 3170-3177

Mallipeddi R, Suganthan PN, Pan QK, Tasgetiren MF (2011). Differential evolution algorithm with ensemble of parameters and mutation strategies. Appl. Soft Comput. 11(2): 1679–1696

Massa A, Pastorino M, Randazzo A (2006). Optimization of the directivity of a monopulse antenna with a subarray weighting by a hybrid differential evolution method. IEEE Antennas and Wireless Propagation Lett. 5(1): 155–158

Peng H, Wu Z (2015). Heterozygous differential evolution with Taguchi local search. Soft Computing 19(11): 3273-3291

Peng H, Wu Z, Shao P, Deng C (2016). Dichotomous binary differential evolution for knapsack problems. Mathematical Problems in Engineering 2016

Peng H, Guo Z, Deng C, Wu Z (2017). Enhancing differential evolution with random neighbors based strategy. Journal of Computational Science 2017

Price K, Storn RM, Lampinen JA (2006). Differential evolution: a practical approach to global optimization. Springer Science & Business Media

Qin AK, Huang VL, Suganthan PN (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. Evol. Comput. 13(2):398–417

Qing A (2006). Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems. IEEE Trans. Geoscience and Remote Sensing 44(1): 116–125

Rahnamayan S, Tizhoosh HR, Salama M (2008). Opposition-based differential evolution. IEEE Trans. Evol. Comput. 12(1): 64–79

Segura C, Coello CAC, Segredo E, Miranda G, León C (2013). Improving the diversity preservation of multi-objective approaches used for single-objective optimization. In Proc. 2013 IEEE Congr. Evol. Comput. pp. 3198–3205

Segura C, Coello CAC, Hernández-Díaz AG (2015). Improving the vector generation strategy of differential evolution for large-scale optimization. Inform. Sci. 323: 106–129

Storn R, Price K (1995). Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. vol. 3, ICSI Berkeley

Tagawa K (2009). Survival selection methods for the differential evolution based on continuous generation model. In Proceedings of International Symposium on Autonomous Decentralized Systems pp. 1–6

Tanabe R, Fukunaga AS (2013). Success-history based parameter adaptation for differential evolution. In Proceedings of 2013 IEEE Congr. Evol. Comput. pp. 71–78

Tanabe R, Fukunaga AS (2014). Improving the search performance of shade using linear population size reduction, In Proceedings of 2014 IEEE Congr. Evol. Comput. pp. 1658–1665

Wang H, Rahnamayan S, Sun H, Omran MG (2013). Gaussian bare-bones differential evolution. IEEE Trans. Cybern. 43(2): 634–647

Wang Y, Cai Z, Zhang Q (2011), Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. Evol. Comput. 15(1): 55–66

Wang Y, Li HX, Huang T, Li L (2014). Differential evolution based on covariance matrix learning and bimodal distribution parameter setting. Appl. Soft Comput. 18: 232-247

Wang Y, Liu ZZ, Li J, Li HX, Yen GG (2016). Utilizing cumulative population distribution information in differential evolution. Appl. Soft Comput. 48: 329-346

Wang Y, Liu ZZ, Li J, Li HX, Wang J (2016). On the selection of solutions for mutation in differential evolution. Frontiers of Computer Science 1-19

Yao X, Liu Y, Lin G (1999), Evolutionary programming made faster. IEEE Trans. Evol. Comput. 3(2):82–102

Yildiz AR (2013). A new hybrid differential evolution algorithm for the selection of optimal machining parameters in milling operations. Appl. Soft Comput. 13(3): 1561–1566

Yu WJ, Shen M, Chen WN, Zhan ZH, Gong YJ, Lin Y, Liu O, Zhang J (2014). Differential evolution with two-level parameter adaptation. IEEE Trans. Cybern. 44(7): 1080–1099

Zhang J, Sanderson AC (2009), JADE: adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comput. 13(5): 945–958