Differential Evolution with Landscape-Based Operator Selection for Solving Numerical Optimization Problems

Karam M. Sallam, Saber M. Elsayed, Ruhul A. Sarker and Daryl L. Essam

Abstract In this paper, a new differential evolution framework is proposed. In it, the best-performing differential evolution mutation strategy, from a given set, is dynamically determined based on a problem's landscape, as well as the performance history of each operator. The performance of the proposed algorithm has been tested on a set of 30 unconstrained single objective real-parameter optimization problems. The experimental results show that the proposed algorithm is capable of producing good solutions that are clearly better than those obtained from a set of considered state-of-the-art algorithms.

1 Introduction

Optimization is an important decision making tool in many fields, including, but not limited to, operations research, engineering design and data mining. Without loss of generality, a global unconstrained single objective optimization problem, as considered in this paper, can be stated as finding the values of a decision vector $\vec{x} = (x_1, x_2, \ldots, x_D) \in \mathbb{R}^{\mathrm{D}}$, which satisfies the variable bounds, $x^{min} \leq x \leq x^{max}$ and minimizes or maximizes an objective function $f(\vec{x})$, where x^{min} and x^{max} are the lower and upper boundaries, respectively. In these problems, the decision variables may be integer, real, discrete, or mixed [10] and the objective function can be linear or

K.M. Sallam (✉) · S.M. Elsayed · R.A. Sarker · D.L. Essam
School of Engineering and Information Technology, University of New South Wales, Canberra, Australia
e-mail: karam.sallam@student.adfa.edu.au

S.M. Elsayed
e-mail: s.elsayed@adfa.edu.au

R.A. Sarker
e-mail: r.sarker@adfa.edu.au

D.L. Essam
e-mail: d.essam@adfa.edu.au

G. Leu et al. (eds.), *Intelligent and Evolutionary Systems*, Proceedings in Adaptation, Learning and Optimization 8,
DOI 10.1007/978-3-319-49049-6_27

nonlinear, convex or non-convex, continuous or not continuous, and uni-modal or multi-modal [9].

As gradient based methods usually encounter many difficulties when solving such complex problems [16], evolutionary algorithms (EAs) have received much interest over the last few decades. EAs are population-based search strategies that have demonstrated promising results in solving complex optimization problems [29]. The reasons for this popularity are (1) they do not require the satisfaction of specific mathematical properties; (2) they are flexible to dynamic changes; and (3) they have the capability for self-organization [12]. However, as EAs are stochastic algorithms, there is no guarantee that they will reach an optimal solution in every run. To add to this, the performance of EAs depends on parameter settings.

The family of EAs contains various algorithms, such as differential evolution (DE) [30], genetic algorithm (GA) [13] and evolution strategy (ES) [27]. The major difference between these algorithms, is in the way they produce new solutions. Among those algorithms, DE has gained popularity in solving continuous optimization problems [7, 28]. However, there is no guarantee that a DE algorithm, which performs well for one problem, or a certain class of problems, will work well for another, or on a range of problems. One reason for this is the variability of the underlying mathematical properties of optimization problems.

As a consequence, researchers have proposed multi-operator and multi-method based algorithms to solve complex optimization problems [9, 11]. However, the way of combining these operators and/or methods in the best way is still a challenging task. In the evolutionary algorithms, the selection of operators for use in a search process is made based on different criteria, such as the improvement in the quality of solutions, and/or constraint violations and/or the feasibility rate [9], re-enforcement learning mechanisms [1, 17], convergence differences and progress ratios [14]. However, the use of landscape information in the selection process is rare, even though it may boost the performance of an algorithm if it is carefully incorporated [2, 6]. However, for these methods that do exist, they have some limitations: (1) the landscape analysis was performed using an off-line mode, i.e., initial experiments were conducted to calculate landscape statistics values independently of the evolutionary process used for solving the problem [22, 23]; (2) the calculation of the landscape measures was computationally expensive [23]; and (3) a training and testing mechanism is used, which may mean the algorithm is biased towards the considered test problems, and hence its performance can deteriorate when solving another set of problems.

In this paper, a new DE framework is proposed, in which a function's landscape information is considered, in addition to the usual performance history of the operators in selecting the best-performing DE operator during the evolutionary process. We also consider linear population size reduction, in which population size is reduced continuously with a linear function. In linear population size reduction, the worst individual is deleted to resize the population. In this paper, before deleting the worst-ranking individuals, a modified technique is used, the 2 worst solutions and the centroid of the entire population are used to generate a new individual. If the new one is better than the second worst one, it replaces it. To speed up the convergence

of the proposed algorithm, the sequential quadratic programming (SQP) technique is periodically applied, once every predefined number of generations. This DE algorithm with landscape based operator selection is named DE-LOS.

To judge the performance of the proposed framework, a total of 30 test functions were solved from the CEC2014 competition [18]. These benchmark sets have different mathematical properties, and are of 10, 30, 50 and 100 dimensions. The computational results show that the performance of DE-LOS is much better than the top two algorithms from the CEC2014 competition.

The rest of this paper is organized as follows: in Sect. 2, a review of DE algorithms and operators are reviewed, along with some landscape measures. Section 3 presents the proposed framework. The simulation results on benchmark problems, and the value of parameters are provided in Sect. 4. Finally, Sect. 5 provides conclusions and possible future research directions.

2 Related Work

In this section, a literature review of DE and the concept of landscape analysis are discussed.

2.1 Differential Evolution Algorithm

DE was proposed by Storn and Price [30]. It is a popular EA because it usually converges fast, is simple in implementation, and the same settings can be used for many different optimization problems. As of the literature, DE showed good performance in comparison to several other EAs on a wide variety of problems [8]. The DE algorithm uses three operators (mutation, crossover and selection) to evolve a population of individuals during the search process.

2.2 Improved DE Algorithms

In this section, some of the improved variants of DE are discussed.

2.2.1 Single Operator de Variants

An adaptive DE algorithm with an optional external memory (JADE) was proposed by Zhang et al. [35], in which the CR_i of each individual x_i at each generation was independently generated according to a normal distribution of mean μCr and standard deviation 0.1, where when the value of CR_i falls outside [0,1], it is repaired to a

value in [0,1]. Also, the value of, F_i, of each individual, x_i, was independently generated according to a Cauchy distribution with parameter μF and scale parameter 0.1. If its value is greater than 1, then it is truncated to 1, or regenerated if $F_i < 0$.

Success-history based parameter adaptation for differential evolution (SHADE), which is an improved version of JADE, uses a history based parameter adaptation method. In SHADE, instead of using a single pair (μCR, μF) to guide parameter adaptation, the mean values of *SCR* and *SF* for each generation, were stored in memory as *MCR* and *MF*.

The L-SHADE [31] algorithm is a SHADE algorithm that uses linear population size reduction (LPSR) to dynamically re-size its population during a run. LPSR reduces the population linearly as the number of fitness evaluations increases. LSHADE showed good performance, in comparison with other algorithms over a set of unconstrained optimization problems.

Sallam et al. [28] proposed a neurodynamic differential evolution algorithm for solving the CEC2015 single objective optimization problems. An adaptive mechanism was proposed for the appropriate use of LSHADE and neuro-dynamic during the search process.

2.2.2 Multi-operator DE Variants

In this section, a brief review of multi-operator based DE and self-adaptive DE is provided.

Self adaptive multi-operator differential evolution (SAMO-DE) was proposed by Elsayed et al. [9] for solving constrained optimization problems. In their proposed algorithm, each operator has its own sub-population which are evolved by different DE operators. Based on an improvement measure, in which the solution quality, constraint violation and feasibility ratio were used to calculate the success of each operator, the number of individuals in each sub-population was adaptively updated, and more emphasis was given to the operator with the highest success. The results showed that SAMO-DE performed better than other-state-of-the-art algorithms.

Composite DE (CoDE) was proposed by Wang et al. [33] for solving optimization problems. In CoDE, three mutation strategies were randomly combined with three fixed control parameter settings for generating a new trial vector at each generation. To generate a new solution, three vectors were generated, then the best one among them was selected to enter the next generation. From the experimental results, it was concluded that CoDE is a promising DE algorithm for solving optimization problems.

A self-adaptive DE (SaDE) was proposed by Qin et al. [26] for solving unconstrained real-parameter optimization. In SaDE, both the trial vector generation strategy and its associated control parameter values, were gradually self-adapted according to a success rate, that was calculated based on previous learning experience. At the beginning, all mutation strategies had equal probability to generate a new solution, and the probability was updated after an initial *LP* generations, accordingly as follows: at the end of each generation, after evaluating all the generated trial vectors,

the number of trial vectors generated by each strategy that successfully entered the next generation was recorded in its success memory and the number of trial vectors generated by each strategy that failed to enter the next generation was recorded in its failure memory. This algorithm performed much better than both the traditional DE algorithm and several state-of-the-art adaptive parameter DE variants.

All of the above mentioned methods did not incorporate any landscape information in the selection phase.

2.3 Landscape Analysis

Generally, a fitness landscape consists of: (1) a set of solutions (populations of individuals), (2) fitness values (objective function values) of individuals, and (3) a neighborhood operator which can be used as a distance measure [19, 22]. Measuring the fitness landscape of a problem aids researchers to classify a problem as easy or hard to solve [25]. Many landscape measures have been proposed to understand and analyze different characteristics of a problem [19, 24], and this section reviews some of them.

Auto-correlation is often used to measure the ruggedness of a fitness landscape [5, 24]. Fitness distance correlation (FDC), proposed by Jones and Forrest [15], is another method used to measure problem difficulty [32]. It measures the correlation between the objective value and the distance to the nearest optimum in the search domain. Among landscape measures is also the searchability of a problem. To measure the searchability of a problem, which is the ability of the search operator to move to a region of a search space of better fitness value, an information landscape metric exists, which is computed based on the difference between the information landscape vector of the problem to be solved and a reference landscape vector. The reference landscape is the landscape of a function that is easy to be optimized by any optimization algorithm in any dimension [3].

An information matrix $M = [a_{i,j}]$ for a minimization problem, is constructed using Eq. 1

$$a_{i,j} = \begin{cases} 1 & \text{if } f(x_i) < f(x_j) \\ 0.5 & \text{if } f(x_i) = f(x_j) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Not all of the entries in the information landscape are necessary for defining the information landscape [3, 4]. There is duplication in the entries due to symmetry (so the lower triangle should be omitted), the entries on the diagonal are always 0.5 (and also should be omitted), and the row and column of the optimum solution should also be omitted. So, the information matrix can be reduced to a vector $LS = (ls1, ls_2, ..., ls_{|LS|})$, where the number of elements in LS, $|LS| = \frac{(NP-1)\times(NP-2)}{2}$. Continuing from this:

$$LD = \frac{1}{|LS|} \times \sum_{i=1}^{|LS|} |(ls_i)_f - (ls_i)_p| \tag{2}$$

where $(ls_i)_p$ is the information landscape vector of the problem to be solved, and $(ls_i)_f$ is the information landscape vector of the reference function. When LD is near 0, the problem is considered easy, while $LD = 1$, means the problem is difficult.

In the recent past, researchers and practitioners have used fitness landscape to determine and select an appropriate algorithm or operator for solving optimization problems. In [20], a prediction model was developed to predict when a particle swarm optimization (PSO) algorithm would fail to solve a particular optimization problem. Decision trees were employed to predict the failure of seven different PSO algorithms, by using a number of different fitness landscape metrics. In [6], an adaptive operator selection mechanism, based on a set of four fitness landscape analysis techniques, was used to train an online regression learning model (dynamic weighted majority), which was used to predict the weight of each operator in each generation. Their proposed mechanism was used to determine the most suitable crossover operator, among four crossover operators, to solve a set of Capacitated Arc Routing Problem (CARP) instances. The authors used instantaneous reward, in which the reward was considered as the value computed at the last evaluation. In comparison with some of the-state-of-the-art algorithms, the algorithm did not show significant benefit.

3 Landscape-Based Adaptive Operator Selection DE

In this section, our novel DE-LOS algorithm is presented.

3.1 DE-LOS

The existing multi-operator algorithms use an adaptive operator selection mechanism, which is usually based on the success of generating new offspring. In this section, a DE-LOS algorithm is proposed, which uses problem landscape information, as well as the performance of operators, to adaptively place emphasis on the most suitable DE operator. The general steps in DE-LOS are given in Algorithm 1.

To begin with, three mutation strategies (DE/φbest/1, DE/current-to-φbest/1/archive and DE/current-to-φbest/1/without archive) are used. Initially, NP random individuals are generated within the variable bounds using a Latin Hypercube design. Then, each operator is randomly assigned to the same number of individuals. Next, a new solution is generated using its assigned mutation strategy. At the same time, the information landscape negative searchability metric and performance history, using Eqs. 4 and 2, respectively, are calculated for each single operator. This process con-

Algorithm 1 Proposed algorithm

```
1: C ← 0; Generate an initial population (X) of size NP using Latin Hypercube Design;
2: Calculate the fitness values of X;
3: FES ← FES + NP;
4: while FES ≤ MAX_FES do
5:     C ← C + 1;
6:     if FES ≤ limit then
7:         if C < CS then
8:             Evolve the population using m DE operators;
9:         else if C ≥ CS and C < 3CS then
10:            if mod(C, CS==0) then
11:                Calculate the average normalized value ANV for each DE operator using Eq. 7;
12:                m ← m − 1 - i.e., the best m − 1 DE operators;
13:                Evolve the population using the best m DE operators;
14:            else if mod(C, 2CS==0) then
15:                Calculate average normalized value ANV of each DE operator using Eq. 7;
16:                m = m − 1, i.e., discard the worst DE operator;
17:            end if
18:            Evolve the population using m operators;
19:        else if mod(C, 3CS==0) then
20:            C ← 0, reset m to 3, and go to step 5;
21:        end if
22:    else
23:        if mod(iter,100==0) then
24:            apply SQP as a local search up to a fixed number of fitness evaluations.
25:        end if
26:        Evolve the population using the best DE operator;
27:    end if
28:    FES ← FES + NP
29:    Update population using Eq. 8
30: end while
```

tinues for a certain number of generations, say CS generations. After CS generations, the average value of the landscape metric and performance history are computed for every operator, using Eqs. 5 and 6, respectively. Subsequently, the normalized value of both measures is computed using Eq. 7. Based on this value, the best two operators are selected to be used in the subsequent cycle. Throughout the next cycle, at each generation, offspring are generated using one of those two operators, while the performance measure and landscape value are calculated for each operator. Then, the normalized values are calculated for the two mutation strategies. Based on the overall mean normalized performance measure (Eq. 7), the worst operator (the one with the minimum value) is discarded. Subsequently, the remaining best operator is used to evolve the entire population, for the subsequent CS generations. Note that after every CS generations, the success and landscape metrics are reset to zero. The above process is repeated every $3CS$ generations, however, after a predefined number of fitness evaluations is reached, the best-performing operator so far, is used to evolve the population until a stopping criterion is reached. Furthermore, during this stage, SQP is periodically applied to the best individual from the whole population.

Algorithm 2 Algorithm for computing the Information landscape negative searchability.

1: Input: population of individuals (X) updated by operator op;
2: Determine the location of the best individual in the sample, x^*.
3: Construct the pairwise comparison matrix M using Eq. 1;
4: Construct vector LS_f that represents the information matrix of the problem.
5: Construct the reference function, f_{ref}, by using Eq. 3.
6: Construct the vector LS_{ref} that represents the information landscape of the reference function.
7: Compute the value of the Information Landscape negative searchability index using Eq. 2.

3.2 The Selection Phase

3.2.1 Information Landscape Negative Searchability Measure

The information landscape negative searchability measure, which is based on the difference between the information landscape vector of the problem to be solved and a well-known spherical function as a reference landscape, is considered in this research, due to its simplicity and scalability [21].

The reference function $f_{ref}(\vec{x})$ is constructed using Eq. 3.

$$f_{ref}(\vec{x}) = \sum_{j=1}^{D} (x_j - x_j^*)^2 \tag{3}$$

where $\vec{x}_i^*$ is the best individual in the sample.

In this paper, Latin Hypercube Design is used to generate an initial population [34] that properly covers the search space of the problem. After constructing the vector landscape of the problem to be optimized (LS_f) and the vector landscape of the reference function (LS_{ref}), the information landscape negative searchability measure is computed using Eq. 2, this is done as part of Algorithm 2.

3.2.2 Average Normalized Value (ANV)

After the information landscape negative searchability value for each operator was computed, the success rate (SR) of each operator is computed. The success rate of each operator (SR_{op}) is defined as the number of successful offspring generated by a search operator (op), divided by the number of individuals assigned to op, as shown in Eq. 4:

$$SR_{op} = \frac{\text{Number of improved offsprings}}{\text{Number of all individuals evolved by operator}} \tag{4}$$

The normalized value for the *SR* and landscape metrics are calculated using Eqs. 5 and 6, respectively.

$$NM_{SR} = \frac{M_{SR_{OP}}}{\sum_{OP=1}^{m} M_{SR_{OP}}} \tag{5}$$

$$NM_{LD} = \frac{(1 - M_{LD_{OP}})}{\sum_{OP=1}^{m} (1 - M_{LD_{OP}})} \tag{6}$$

where M_{SR} and M_{LD} are the mean value of the success rate and landscape value, respectively.

Subsequently, the normalized performance of each operator is computed using Eq. 7:

$$ANV_{OP} = (NM_{SR_{OP}} + NM_{LD})/2 \tag{7}$$

3.3 Population Updating Method

A linear population size reduction scheme is used to adaptively re-size *NP* during the evolutionary process [31], as follows:

$$NP_{iter} = round[(\frac{NP^{min} - NP^{max}}{FES_{max}}) \times cfe + NP^{max}] \tag{8}$$

where NP^{min} is the smallest number of individuals that the proposed algorithm can use. *cfe* is the current number of fitness evaluations, FES_{max} is the maximum number of fitness evaluations. The default value of NP^{max} is set as $18D$, NP^{min} is set as 7.

To get some benefit from the worst individuals before deleting them, a new solution is generated using information from the worst two individuals and the centroid of the population ($X_{cent} = \frac{\sum_{j=1}^{D} \sum_{i=1}^{NP} x_{i,j}}{NP}$), as

$$X_{new} = X_{cent} + rand \times (X_{NP} - X_{NP-1}) \tag{9}$$

Then the worst individual is deleted, and a decision is made to decide if X_{NP-1} is replaced by X_{new} or not, based on the objective value.

4 Experimental Results

In this section, the performance of the proposed algorithm is tested by solving a set of problems taken from the CEC2014 competition on learning-based real-parameter single objective optimization [18]. The CEC2014 benchmark test set contains 30 test

problems. The search space for all the problems is $[-100, 100]^D$. The proposed algorithm was run following the guidelines of the competition. That required 51 independent runs for each test problem with up to $FES_{MAX} = 10,000$D fitness evaluations. In the experimentation, if the deviation of the best fitness value from the optimal solution is less than or equal to $1.0e-8$, it was considered as zero. The algorithm was coded using Matlab R2014a, and was run on a PC with a 3.4 GHz Core I7 processor with 16 GB RAM, and windows 7.

4.1 Algorithm Parameters and Operators

The default values of NP^{init}, and NP^{min} were set based on our experimental analysis, $NP^{init} = 18D$ and $NP^{min} = 7$. φ was set at a value of 0.6 for DE/φbest/1 to maintain diversity, while its value was 0.1, for the other two variants, to speed up the convergence rate. A is the archive rate, and it was set at a value of 1.4. H, the memory size, was set at the value of 5. *limit* the maximum limit to run the multi-operator phase, where as after it the best performing operator evolves the population until the end of the run, was set at the value of $\frac{2}{3} \times FES_{MAX}$, and CS was100. The scaling factor F and the crossover probability CR were set as in [31].

4.2 Detailed Results for 10, 30, and 50D

The computational results of DE-LOS for 10, 30, and 50D are shown in Table 1. For 10D, from the results obtained, the proposed algorithm provided the optimal solutions for all unimodal functions ($F01 - F03$). For the multimodal functions ($F04 - F16$), DE-LOS was able to obtain the optimal solutions on six problems, while it was very close for the rest. For hybrid functions ($F17 - F22$), DE-LOS was able to obtain the optimal solution for only $F17$, and was very close for the rest of the test problems. However it became stuck in local solutions for all the composition test problems, $F23 - F30$.

For 30D, from the results, DE-LOS was able to obtain the optimal solution on all the unimodal problems. For multimodal problems, DE-LOS was able to obtain the optimal solution for $F04$, $F06$, $F07$, $F08$ and $F10$, while it was very close to the optimal solution for the rest. For hybrid functions, the best solutions obtained were close to the optimal. Again, for the composition problems, DE-LOS got stuck in local solutions.

Table 1 Detailed Results for 10D

	10D			30D			50D		
	Best	Mean	Std.	Best	Mean	Std.	Best	Mean	Std.
F01	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	1.4941E-02	9.7381E+00	1.6581E+01
F02	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F03	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F04	**0.0000E+00**	2.5318E+01	1.5547E+01	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	3.2701E+01	4.6706E+01
F05	2.1219E-05	1.4533E+01	8.2010E+00	1.9999E+01	2.0000E+01	8.4719E-05	2.0000E+01	2.0000E+01	3.0879E-04
F06	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	2.2540E-02	9.4797E-02	2.1569E-04	6.6749E-01	8.4004E-01
F07	**0.0000E+00**	8.7003E-04	2.6924E-03	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	1.9342E-04	1.3813E-03
F08	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**	**0.0000E+00**
F09	**0.0000E+00**	2.0289E+00	2.1428E+00	1.9899E+00	1.1179E+01	7.5265E+00	1.0945E+01	2.2215E+01	8.0015E+00
F10	**0.0000E+00**	1.2246E-03	8.7454E-03	0.0000E+00	1.6329E-03	7.0209E-03	0.0000E+00	5.4217E-02	6.8031E-02
F11	1.8736E-01	4.5646E+01	7.0302E+01	7.2417E+02	1.2808E+03	2.1279E+02	2.0309E+03	3.2116E+03	4.4592E+02
F12	0.0000E+00	4.5128E-02	3.5151E-02	2.1114E-02	6.5626E-02	3.0115E-02	1.7739E-02	6.0733E-02	2.5329E-02
F13	1.9079E-02	4.4725E-02	1.2332E-02	8.9616E-02	1.2119E-01	1.8950E-02	1.3417E-01	1.7983E-01	2.1522E-02
F14	2.0373E-02	8.6169E-02	2.8643E-02	1.4846E-01	2.2851E-01	2.5230E-02	2.7022E-01	2.9956E-01	1.8229E-02
F15	2.4961E-01	5.1461E-01	1.7170E-01	1.3835E+00	1.9349E+00	2.9087E-01	2.9296E+00	4.5685E+00	8.5283E-01
F16	2.8912E-01	1.1718E+00	3.6937E-01	7.5888E+00	8.8845E+00	5.2729E-01	1.6029E+01	1.7271E+01	6.9797E-01
F17	**0.0000E+00**	1.0520E+00	1.7762E+00	4.4058E+01	1.4910E+02	8.3895E+01	4.2814E+02	1.0661E+03	3.7011E+02
F18	6.8422E-04	8.7598E-02	1.3293E-01	1.2766E+00	5.4211E+00	2.4883E+00	4.9062E+01	8.6760E+01	1.2591E+01
F19	2.5697E-02	9.0563E-02	7.8481E-02	1.1901E+00	2.9193E+00	6.6671E-01	6.1461E+00	8.9485E+00	1.8277E+00
F20	8.8738E-04	1.7272E-01	1.5260E-01	9.0747E-01	3.4603E+00	1.3381E+00	5.7622E+00	1.2356E+01	3.8644E+00

(continued)

Table 1 (continued)

	10D			30D			50D		
	Best	Mean	Std.	Best	Mean	Std.	Best	Mean	Std.
F21	1.2221E-04	4.0420E-01	3.2446E-01	1.5087E+00	7.9776E+01	7.0129E+01	2.4379E+02	4.2392E+02	9.4926E+01
F22	1.7450E-02	6.0789E-02	3.8817E-02	1.1833E+01	2.4106E+01	5.9125E+00	2.7924E+01	1.4015E+02	9.1661E+01
F23	2.0000E+02	3.2438E+02	2.5379E+01	2.0000E+02	3.0688E+02	2.9265E+01	2.7186E+02	3.4259E+02	1.0102E+01
F24	1.0000E+02	1.0976E+02	2.2786E+00	2.0000E+02	2.1692E+02	1.0554E+01	2.0000E+02	2.5604E+02	3.1495E+01
F25	1.0000E+02	1.1584E+02	1.4053E+01	2.0000E+02	2.0240E+02	7.0748E-01	2.0000E+02	2.0497E+02	1.2932E+00
F26	1.0001E+02	1.0004E+02	1.6358E-02	1.0008E+02	1.0012E+02	1.5014E-02	1.0011E+02	1.0016E+02	1.8533E-02
F27	8.1331E-01	2.1034E+01	7.9706E+01	2.0000E+02	3.0000E+02	2.0000E+01	3.0023E+02	3.3037E+02	3.1344E+01
F28	2.0000E+02	3.6736E+02	4.5559E+01	2.0000E+02	8.1699E+02	1.1047E+02	2.0000E+02	1.0982E+03	1.3089E+02
F29	1.3175E+02	2.2014E+02	1.2629E+01	7.1325E+02	7.1643E+02	3.7186E+00	6.9851E+02	8.0150E+02	3.9877E+01
F30	4.5429E+02	4.6307E+02	5.7407E+00	4.0223E+02	6.6248E+02	2.4821E+02	8.0234E+03	8.8013E+03	5.6733E+02

For 50D, DE-LOS was able to obtain the optimal solutions in $F02$ and F03, while for $F01$ it obtained very close solutions to the optimal one. For multimodal problems, DE-LOS was robust in solving $F08$, efficient in solving $F04$, $F07$, $F08$ and $F10$, while it got stuck in local solutions for the rest of the test problems. This was also the situation for the hybrid and composition problems, although its performance in solving the hybrid problems was a little bit better than its performance in solving the composition problems.

4.3 DE-LOS Versus State-of-the-art Algorithms on CEC2014

DE-LOS was compared with the top two algorithms in the literature LSHADE [31] and UMOEAs [11]. The matlab source codes for LSHADE and UMOEAs were downloaded online. We ran these algorithms using the same parameters suggested by the authors in their papers and the other conditions were the same as the competition guidelines. To make a fair comparison, all the algorithms were run using the same seeds.

Table 2 shows a comparison summary of the results obtained from DE-LOS and the other two algorithms for 10D, 30D, and 50D problems. A non-parametric test, Wilcoxon rank-sum test, was chosen, to judge the difference between any paired algorithms. The results regarding the best and average fitness functions are presented in Table 2. The significance level was set at a value of 10 %. Based on the test results/rankings, one of three signs (+, −, and ≈) was assigned for the comparison of any two algorithms (shown in the last column), where the "+" sign means that the first algorithm is significantly better than the second, the "−" sign means that the first algorithm is significantly worse, and the "≈" sign means that there is no significant difference between the two algorithms. Considering the quality of solutions, and from the results in Table 2, it is clear that DE-LOS is always better than the other algorithms, based on the best and average results obtained, and this is obvious for 30D and 50D.

Based on the statistical test, DE-LOS is better than UMOEAs in 10D, 30D, and 50D in regard to best and average results, except for the best results in 10D and 50D, where there is no significant difference between DE-LOS and UMOEAs. Considering the comparison between DE-LOS and LSHADE, DE-LOS is significantly better than LSHADE in 10D, 30D, and 50D.

In addition, based on the average results obtained, the average ranking of DE-LOS, LSHADE and UMOEAs, as produced by the Friedman test, is summarized in Table 3. The results in Table 3 are consistent with the results in Table 2, in which DE-LOS had the best rank.

Table 2 A comparison summary between DE-LOS and other state-of-the-art algorithms

	10D					30D				50D			
DE-LOS		Better	Equal	Worse	Dec.	Better	Equal	Worse	Dec.	Better	Equal	Worse	Dec.
Versus LSHADE	Best	14	10	6	+	17	9	4	+	18	4	8	+
	Mean	18	6	6	+	14	8	8	+	16	10	4	+
Versus UMOEAs	Best	10	13	7	≈	15	8	7	+	12	6	12	≈
	Mean	18	4	8	+	22	4	4	+	24	1	5	+

Table 3 Friedman's test results

Algorithm	10D	30D	50D
	Rank	Rank	Rank
DE-LOS	**1.63**	**1.60**	**1.58**
LSHADE	2.08	1.85	1.85
UMOEAs	2.28	2.55	2.57

5 Conclusion and Future Work

During the last few decades, DE algorithms have shown superior performance to many other-state-of-the-art algorithms in solving both unconstrained and constrained optimization problems. It is known that no single algorithm or operator is able to solve all kinds of optimization problems. Even though for a single run, an algorithm or operator may perform well in the earlier generations, its performance often decreases during later generations. So the selection of an appropriate algorithm or operator is not an easy task. In this paper, the DE-LOS algorithm has been presented. It used landscape and normalized performance measures to dynamically place more emphasis of the best-performing DE mutation.

The algorithm has been tested on 30 bound constrained numerical optimization problems from the CEC2014 competition. The results obtained were better than those obtained from the best two algorithms in the literature.

In future work, we will investigate the use of more than one landscape measure, and will incorporate some of them with multi-method-based algorithms.

References

1. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning 47(2-3), 235–256 (2002)
2. Bischl, B., Mersmann, O., Trautmann, H., Preuß, M.: Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In: Proceedings of the 14th annual conference on Genetic and evolutionary computation. pp. 313–320. ACM (2012)
3. Borenstein, Y., Poli, R.: Information landscapes. In: Proceedings of the 7th annual conference on Genetic and evolutionary computation. pp. 1515–1522. ACM (2005)
4. Borenstein, Y., Poli, R.: Decomposition of fitness functions in random heuristic search. In: Foundations of Genetic Algorithms, pp. 123–137. Springer (2007)
5. Chicano, F., Luque, G., Alba, E.: Autocorrelation measures for the quadratic assignment problem. Applied Mathematics Letters 25(4), 698–705 (2012)
6. Consoli, P.A., Minku, L.L., Yao, X.: Dynamic selection of evolutionary algorithm operators based on online learning and fitness landscape metrics. In: Simulated Evolution and Learning, pp. 359–370. Springer (2014)
7. Das, S., Suganthan, P.N.: Differential evolution: a survey of the state-of-the-art. Evolutionary Computation, IEEE Transactions on 15(1), 4–31 (2011)

8. Elsayed, S.M., Sarker, R.A., Essam, D.L.: Differential evolution with multiple strategies for solving cec2011 real-world numerical optimization problems. In: Evolutionary Computation (CEC), 2011 IEEE Congress on. pp. 1041–1048. IEEE (2011)
9. Elsayed, S.M., Sarker, R.A., Essam, D.L.: Multi-operator based evolutionary algorithms for solving constrained optimization problems. Computers & operations research 38(12), 1877–1896 (2011)
10. Elsayed, S.M., Sarker, R.A., Essam, D.L.: Memetic multi-topology particle swarm optimizer for constrained optimization. In: Evolutionary Computation (CEC), 2012 IEEE Congress on. pp. 1–8. IEEE (2012)
11. Elsayed, S.M., Sarker, R.A., Essam, D.L., Hamza, N.M.: Testing united multi-operator evolutionary algorithms on the cec2014 real-parameter numerical optimization. In: Evolutionary Computation (CEC), 2014 IEEE Congress on. pp. 1650–1657. IEEE (2014)
12. Fogel, L.J., Owens, A.J., Walsh, M.J.: Artificial intelligence through simulated evolution (1966)
13. Goldberg, D.E., Holland, J.H.: Genetic algorithms and machine learning. Machine learning 3(2), 95–99 (1988)
14. Gordián-Rivera, L.A., Mezura-Montes, E.: A combination of specialized differential evolution variants for constrained optimization. In: Advances in Artificial Intelligence–IBERAMIA 2012, pp. 261–270. Springer (2012)
15. Jones, T., Forrest, S., et al.: Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In: ICGA. vol. 95, pp. 184–192 (1995)
16. K. Deb: Optimization for engineering design: Algorithms and examples. PHI Learning Pvt. Ltd. (2012)
17. Li, K., Fialho, A., Kwong, S., Zhang, Q.: Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. Evolutionary Computation, IEEE Transactions on 18(1), 114–130 (2014)
18. Liang, J., Qu, B., Suganthan, P.: Problem definitions and evaluation criteria for the cec 2014 special session and competition on single objective real-parameter numerical optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore (2013)
19. Malan, K.M., Engelbrecht, A.P.: A survey of techniques for characterising fitness landscapes and some possible ways forward. Information Sciences 241, 148–163 (2013)
20. Malan, K.M., Engelbrecht, A.P.: Particle swarm optimisation failure prediction based on fitness landscape characteristics. In: Swarm Intelligence (SIS), 2014 IEEE Symposium on. pp. 1–9. IEEE (2014)
21. Malan, K., Engelbrecht, A.: Characterising the searchability of continuous optimisation problems for pso. Swarm Intelligence 8(4), 275–302 (2014)
22. Mersmann, O., Bischl, B., Trautmann, H., Preuss, M., Weihs, C., Rudolph, G.: Exploratory landscape analysis. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. pp. 829–836. ACM (2011)
23. Muñoz, M.A., Kirley, M., Halgamuge, S.K.: A meta-learning prediction model of algorithm performance for continuous optimization problems. In: Parallel Problem Solving from Nature-PPSN XII, pp. 226–235. Springer (2012)
24. Pitzer, E., Affenzeller, M.: A comprehensive survey on fitness landscape analysis. In: Recent Advances in Intelligent Engineering Systems, pp. 161–191. Springer (2012)
25. Poursoltan, S., Neumann, F.: Ruggedness quantifying for constrained continuous fitness landscapes. In: Evolutionary Constrained Optimization, pp. 29–50. Springer (2015)
26. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. Evolutionary Computation, IEEE Transactions on 13(2), 398–417 (2009)
27. Rechenberg, I.: Evolution strategy. Computational Intelligence: Imitating Life 1 (1994)
28. Sallam, K.M., Sarker, R.A., Essam, D.L., Elsayed, S.M.: Neurodynamic differential evolution algorithm and solving cec2015 competition problems. In: Evolutionary Computation (CEC), 2015 IEEE Congress on. pp. 1033–1040. IEEE (2015)

29. Sarker, R., Kamruzzaman, J., Newton, C.: Evolutionary optimization (evopt): a brief review and analysis. International Journal of Computational Intelligence and Applications 3(04), 311–330 (2003)
30. Storn, R., Price, K.: Differential evolution a simple and efficient adaptive scheme for global optimization over continuous spaces, international computer science institute, berkeley. Berkeley, CA (1995)
31. Tanabe, R., Fukunaga, A.S.: Improving the search performance of shade using linear population size reduction. In: Evolutionary Computation (CEC), 2014 IEEE Congress on. pp. 1658–1665. IEEE (2014)
32. Tomassini, M., Vanneschi, L., Collard, P., Clergue, M.: A study of fitness distance correlation as a difficulty measure in genetic programming. Evolutionary Computation 13(2), 213–239 (2005)
33. Wang, Y., Cai, Z., Zhang, Q.: Differential evolution with composite trial vector generation strategies and control parameters. Evolutionary Computation, IEEE Transactions on 15(1), 55–66 (2011)
34. Ye, K.Q.: Orthogonal column latin hypercubes and their application in computer experiments. Journal of the American Statistical Association 93(444), 1430–1439 (1998)
35. Zhang, J., Sanderson, A.C.: Jade: adaptive differential evolution with optional external archive. Evolutionary Computation, IEEE Transactions on 13(5), 945–958 (2009)