Applied Soft Computing xxx (2014) xxx–xxx

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A sinusoidal differential evolution algorithm for numerical optimisation

Q1 Amer Draa*, Samira Bouzoubia, Imene Boukhalfa

MISC Laboratory, Constantine 2 University, Algeria

ARTICLE INFO

Q3 *Article history:*
Received 18 January 2013
Received in revised form 4 November 2014
Accepted 4 November 2014
Available online xxx

Keywords:
Differential evolution
Sinusoidal parameter adjustment
Exploration
Exploitation
Optimisation

ABSTRACT

This paper presents a new variant of the Differential Evolution (DE) algorithm called Sinusoidal Differential Evolution (SinDE). The key idea of the proposed SinDE is the use of new sinusoidal formulas to automatically adjust the values of the DE main parameters: the scaling factor and the crossover rate. The objective of using the proposed sinusoidal formulas is the search for a good balance between the exploration of non visited regions of the search space and the exploitation of the already found good solutions. By applying it on the recently proposed CEC-2013 set of benchmark functions, the proposed approach is statistically compared with the classical DE, the linearly parameter adjusting DE and 10 other state-of-the-art metaheuristics. The obtained results have proven the superiority of the proposed SinDE, it outperformed other approaches especially for multimodal and composition functions.

1. Introduction

Differential Evolution (DE) is a powerful optimisation metaheuristic that was first proposed by Price and Storn in 1995 [1,2]. Since then, this population-based metaheuristic has attracted the attention of researchers in many fields. It was intensively used to solve academic and real life problems, especially in the fields of engineering and sciences. This evolutionary algorithm is considered as one of the most reliable and versatile optimisation techniques available today [3].

Differential evolution uses rather a greedy selection and less stochastic approach to solve optimisation problems than other classical evolutionary algorithms. It differs from other evolutionary algorithms in the mutation and recombination phases. Unlike some metaheuristic techniques such as genetic algorithms and evolution strategies, where perturbation occurs in accordance with random quantities, DE uses weighted differences between solution vectors to perturb a given population [1,4].

According to Price [5], differential evolution has the ability to find the true global optimum regardless of the initial parameter values, is fast and simple with regard to application and modification, requires few control parameters, has a parallel processing nature and offers fast convergence, capable of providing multiple solutions at a single run, effective on integer, discrete and mixed parameter optimisation and able to find the optimal solution for a non-linear constrained optimisation problem with penalty functions. It is clear that conventional DE is a bit overestimated by its inventors; new optimisation problems proved that DE sometimes fails at finding the global optimum, when failing at achieving a good balance between the exploitation of the already-found good solutions and the exploration of the non-visited regions of the search space. This weakness has led to the emergence of many variants of the basic algorithm.

Basic variants of DE were first defined and used by its inventors, Price and Storn, to solve optimisation problems [2,5–7]. Later on, other variants have been proposed and thoroughly studied, by other researchers, to analyse their explorative and exploitive capabilities. For example, in [8], Lampinen proposed a DE algorithm for handling non-linear constraint functions. In [9], Gamperle et al. initiated the series of works studying the influence of parameters' values on the performance of the DE algorithm; the authors gave some guidance about parameter values setting.

Through comparing differential evolution with other evolutionary techniques, such as Particle Swarm Optimisation (PSO) [10], and with the emergence of many new variants of the DE algorithm, the problem of parameters setting and its impact on the overall performance of the algorithm appeared as a serious challenge for researchers interested in exploiting this metaheuristic (DE) for solving daily life problems. Hence, new variants working on the adjustment and adaptation of DE parameters, especially those based on hybridising DE with other evolutionary techniques, have

Q2 * Corresponding author. Tel.: +213 791844353
E-mail addresses: draa_amer@yahoo.fr (A. Draa), samira.bouzoubia@hotmail.fr (S. Bouzoubia), boukhalfaimene@hotmail.com (I. Boukhalfa).

http://dx.doi.org/10.1016/j.asoc.2014.11.003
1568-4946/© 2014 Published by Elsevier B.V.

emerged in recent years [11–16]. A rich review of some of these automatically parameter adjusting and adaptive variants of the DE algorithm can be found in the recent survey of Das and Suganthan about differential evolution [17].

In its original version [1], the DE algorithm uses a special type of mutation in which a given individual is perturbed using the weighted difference between two other individuals chosen randomly from the population. The weight of this difference is known as mutation *scaling factor* or *scaling parameter*. In early generations of the algorithm, the individuals are relatively different from each others. So, small values of the scaling factor will be able to offer good diversity to the population. However, in later generations, the individuals would become very similar to each others, thanks to the crossover operator. So, small values of the scaling factor will be virtually incapable of offering diversity to the populations. Thus, the algorithm is likely to fall in local optima, especially in the case of high-dimension problems, where a big number of generations is generally needed to get good solutions. As a result, more freedom should be allowed to the algorithm to switch from exploration into exploitation, and vice versa, through an automatic adjustment of the values of parameters and so the change of search direction. In the same way, the crossover rate can dramatically influence the quality of solutions; a small value would lead to not getting profit from the mutant, while a big value will make big perturbations.

In this perspective, we propose here a novel automatically parameter adjusting version of differential evolution: the Sinusoidal Differential Evolution (SinDE). This variant uses a sinusoidal framework to define the value of the mutation scaling factor and/or the crossover rate at each generation. Different patterns are proposed; some of them use the sinusoidal adjustment of only one parameter, the other parameter is fixed to the recommended value; other patterns adjust both parameters. This new scheme of calculating DE parameter values, as will be shown in the numerical results section, allows a good balance between exploration and exploitation, since the parameter value increases and decreases periodically, thanks to the periodicity of the Sine function. Moreover, the gradually-increasing (or decreasing) interval between maximum and minimum allowed values of the parameter being adjusted offers a good possibility to escape local optima.

The proposed SinDE algorithm has been tested on the recently defined set of reference benchmark functions: the CEC-2013 testbed [18]. First, It has been compared using the Wilcoxon rank-sum test against the classical DE, the linearly parameter-adjusting DE (LADE), other four recent powerful variants of the DE algorithm (the SMADE, the DEcfbLS, the b6e6rl and the TLBSaDE), and the recently proposed CMA-ES-RIS algorithm. Next, the Holm–Bonferroni [19] statistical procedure has been used to further compare SinDE to these algorithms and five other state-of-the art algorithms. The obtained results show the superiority of the proposed SinDE in both statistics; it has largely outperformed the majority of these algorithms and has been slightly better than the others.

The remainder of this paper is organised as follows. In Section 2, basic concepts of differential evolution and its standard variants are presented. The proposed sinusoidal DE, SinDE, and its different patterns are described in Section 3. Section 4 validates the performances of the proposed approach through comparing it against classical DE and other state-of-the-art metaheuristics. The obtained experimental results are discussed in the same section. Finally, conclusions and future directions of research are drawn up in Section 5.

2. Differential evolution

The Differential Evolution (DE) algorithm [1], follows the general procedure of an evolutionary algorithm: an initial population of individuals is created by random selection and evaluated; then the algorithm enters a loop of generating offspring, evaluating offspring, and selection to create the next generation [20], till a stop condition is met. Generating offspring consists of two operations: differential mutation and differential crossover.

In basic DE, the mutation operation creates a new individual v_i called the *mutant* by adding the weighted difference between two individuals (vectors) chosen randomly from the population to a third one, also chosen randomly, as shown in Eq. (1) below [1]. In the equation, $i, i_1, i_2, i_3 \in \{1, \ldots, NP\}$ are mutually different indices, NP is the size of population, and F is a real positive scaling factor of the difference $d_i = x_{i_2} - x_{i_3}$.

$$v_i = x_{i_1} + F.(x_{i_2} - x_{i_3}) \tag{1}$$

The crossover operator implements a discrete recombination of the mutant, v_i, and the parent vector, x_i, to produce the offspring u_i. This operator is formulated as shown in Eq. (2) below [21,22]; where: $U_j(0, 1)$ is a random number in the interval [0,1], $CR \in [0, 1]$ is the crossover rate, and $k \in \{1, 2, \ldots, d\}$ is a random parameter index, chosen once for each individual i to be sure that at least one parameter is always selected from the mutant. Popular values for CR are in the range [0.4, 1].

$$u_i(j) = \begin{cases} v_i(j), & \text{if } U_j(0, 1) \le CR \text{ or } j = k, \\ x_i(j), & \text{otherwise} \end{cases} \tag{2}$$

This type of crossover is called *binomial crossover*. Another type of crossover used in DE is the *exponential crossover*. Its principle is as follows. The starting position of crossover is chosen randomly from $\{1, \ldots, d\}$, d is the problem dimension; then, L consecutive elements (counted in circular manner) are taken from the mutant vector v [16]. In the present work, we are interested in binomial crossover.

After applying differential crossover, the obtained individual, called *trial vector*, is compared to the parent individual (also called *target vector*). A greedy selection takes place at this point, i.e. the fittest of them will become the target vector. This selection operation, in the case of a minimisation problem, can be expressed as shown in Eq. (3); in the formula, $x_{i,G+1}$ is the new target vector (at generation G+1), $u_{i,G+1}$ is the trial vector obtained from the crossover operation, $x_{i,G}$ is the target vector at generation G, and $fit(*)$ is the fitness function. Of course, in the case of a maximisation problem, the opposite comparison is used, i.e. a '$\geq$' replaces the '$\leq$' in the equation.

$$x_{i,G+1} = \begin{cases} u_{i,G+1}, & \text{if } eval(u_{i,G+1}) \le eval(x_{i,G}), \\ x_{i,G}, & \text{otherwise,} \end{cases} \tag{3}$$

As already stated, the differential evolution algorithm repeats these three operations of mutation, crossover and selection till a stop condition is met. The stop condition is generally chosen to be a predetermined number of generations. Another stop condition that can be used is achieving the global optimum, when its fitness is already known. The non-evolution of the best fitness among population individuals is an other widely-used stop condition. Generally, it is recommended to use a combination of two or three of these stop conditions.

2.1. The 'DE/x/y/z' notation

A number of variations to the basic DE algorithm have been developed in literature, mainly by its inventors [2,5–7]. DE strategies differ in the way the target vector is selected, the number of difference vectors used to perturb it, and haw to determine crossover points. In order to characterise these variations, a notation was adopted in differential evolution literature, namely DE/x/y/z notation [2,7,21]. In this notation, x refers to the method