Artif Intell Rev (2013) 39:195–224
DOI 10.1007/s10462-011-9267-1

A study on scale factor/crossover interaction in distributed differential evolution

Matthieu Weber · Ferrante Neri · Ville Tirronen

Published online: 8 June 2011

Abstract This paper studies the use of multiple scale factor values within distributed Differential Evolution structures employing the so-called exponential crossover. Four different scale factor schemes are proposed, tested, compared and analyzed. Two schemes simply employ multiple scale factor values and two also include an update logic during the evolution. The four schemes have been integrated for comparison within three recently proposed distributed Differential Evolution structures and tested on several various test problems. The results are then compared to those of a previous study where the so-called binomial crossover was employed. Numerical results show that, when associated to the exponential crossover, the employment of multiple scale factors is not systematically beneficial and in some cases even detrimental to the performance of the algorithm. The exponential crossover accentuates the exploitative character of the Differential Evolution, which cannot always be counterbalanced by the increase in the explorative aspect of the algorithm introduced by the employment of multiple scale factor values.

Keywords Differential evolution · Evolutionary algorithms · Distributed algorithms · Scale factor

1 Introduction

Differential Evolution (DE) (see e.g., Price et al. 2005; Storn and Price 1995; Storn 1999) is an evolutionary optimizer, simple yet versatile, which is specially suited for continuous

M. Weber · F. Neri (✉) · V. Tirronen
Department of Mathematical Information Technology, University of Jyväskylä, Agora,
P.O. Box 35, Jyväskylä 40014, Finland
e-mail: ferrante.neri@jyu.fi

M. Weber
e-mail: matthieu.weber@jyu.fi

V. Tirronen
e-mail: ville.tirronen@jyu.fi

optimization problems (see e.g., Chakraborty 2008; Neri and Tirronen 2010; Karaboga and Akay 2009).

DE is population-based metaheuristic (see Price et al. 2005) where, in the original definition of the algorithm, new candidate solutions (called offsprings) are created in a two-step process, often named mutation and crossover. The mutation step consists in creating a provisional offspring by perturbing a parent candidate solution with the weighted difference between two other, randomly chosen candidate solutions (the weight is called the scale factor). Randomly selected variables from the provisional offspring are then exchanged with those of the parent in the crossover step to produce the offspring. The offspring is then evaluated by means of an objective function and compared to the parent. The parent is then replaced by the offspring if the former is outperformed by the latter; otherwise the offspring is discarded.

As explained in Feoktistov (2006), the success of DE as an optimization algorithm is due to the implicit self-adaptation that arises from the algorithm. As described above, the generation of an offspring depends on the distribution in the search space of the other candidate solutions, and more specifically, of the Euclidean distance between pairs of those. In the early stages of the algorithm, the population of those solutions is widely spread over the search space, leading to the creation of provisional offsprings that are geometrically distant from their parent, relatively to the boundaries of the search space: the step size of the search move can be considered as large. However, in the later stages of the optimization process, as the population tends to converge towards specific areas of the search space, the distance between two randomly chosen candidate solutions within these areas decreases and the step size becomes smaller. In other words, due to its structure, a DE scheme is highly explorative at the beginning of the evolution and subsequently becomes more exploitative during the optimization.

The scale factor is obviously one of the parameters of DE that control the convergence by limiting the size of the search step. The number of parent variables that are replaced by variables from the provisional offspring during the crossover step of the algorithm further regulates the distance between a parent and its offspring. This number of variables is determined by the behavior of the crossover scheme, which itself is controlled by another key parameter of DE, the crossover rate. Two common crossover schemes, described in Price et al. (2005), are the binomial crossover and the exponential crossover. While the binomial crossover is dominating in the literature, DE with the exponential crossover has been used as a baseline algorithm in Weber et al. (2010), where it is exhibiting much better performance than the variant with a binomial crossover.

The self-adaptation of DE, although seemingly efficient, conceals a limitation: if the no parent in the population is replaced during the course of a given generation, due to the algorithms inability to produce an offspring that outperforms its parent, the next generation will start with exactly the same parent population, and the optimization process will fall into an undesired stagnation condition (see Lampinen and Zelinka 2000; Neri and Tirronen 2010). Stagnation is that undesired effect which occurs when a population-based algorithm does not converge to a solution (even suboptimal) while the population diversity is still high. In the case of DE, stagnation is defined by the inability of the algorithm, for a prolonged number of generations, to improve upon its population. A closer examination of DE's mutation mechanism reveals that the number of possible exploratory moves is limited by the size of the population; if none of those moves lead to an improvement, the optimization process is compromised. A theoretical analysis of the convergence properties of DE is given in Zaharie (2002a).

The choice of the correct exploratory moves, for a given problem, thus depends on the proper setting of the algorithm's parameters. The influence of the scale factor on the optimization process is very important, as shown e.g., in Zielinski et al. (2006). Several works, e.g. Price and Storn (1997), Storn and Price (1997), Liu and Lampinen (2002b), Liu and Lampinen (2002a), Rönkkönen et al. (2005), have investigated the proper choices in the parameter setting and modern papers, see Gämperle et al. (2002), Liu and Lampinen (2002b), Mallipeddi and Suganthan (2008), have shown that an efficient parameter setting is very prone to problems. Recently, in Tirronen et al. (2009) and Neri et al. (2009), it has been observed that a proper scale factor choice does not depend only on the problem but also on the stage of the optimization process. In other words, to ensure a proper DE functioning, the scale factor must vary over time.

Several works developed various time-varying scale factor mechanisms. According to Das et al. (2005), where a time varying scale factor is, a decrease in the scale factor value during the optimization process can lead to high quality final results. The explanation for success of this mechanism can be seen as a progressive narrowing of the search during evolution and thus an exploitative behavior at the end of optimization with the likelihood of preventing stagnation. Randomization of the scale factor (see e.g., Das et al. 2005) is a popular mechanism for enhancing the DE performance and can be implemented in multiple manners. In Zaharie (2002a), the scale factor is sampled by means of a normal distribution. By following a similar idea, in Rönkkönen and Lampinen (2003), Omran et al. (2005), and Salman et al. (2007), a normal distribution is employed in order to perform the selection of the scale factor. A Cauchy distribution in the self-adaptive scheme is proposed in Soliman et al. (2007) and Soliman and Bui (2008). It is worthwhile mentioning the distinction between dither and jitter, extensively explained in Price et al. (2005): dither is a randomization of the scale factor which samples a scale factor for each candidate solution (vector), the jitter is a randomization of the scale factor for each parameter within each candidate solution. More sophisticated randomization schemes have been proposed in order to integrate a (self-)adaptive logic within the DE frameworks. For example in Brest et al. (2006) a controlled randomization based on uniform distribution is employed. In Neri and Tirronen (2009) the optimal scale factor is periodically determined by means of local search. In Qin et al. (2009) search strategies and parameters are adaptively adjusted during the evolution. In Zhang and Sanderson (2009) the scale factor, as well as other parameters, are sampled from truncated normal distributions. The mean values of these distributions are adapted on the basis of a success rule.

The general trend in the examples above is that the randomization of the scale factor leads in many cases to an improvement in the quality of the final solutions and to a diminution in the risk of stagnation. One possible interpretation to this fact is that a random scale factor increases the possible search moves, allowing to reach solutions that would otherwise be inaccessible, due to DE's inherent limitation in the number of search moves. As a corollary, stagnation conditions may be escaped, the random value of the scale factor allowing an infinite number of potential moves.

A dynamic scale factor is not the only way to improve the performance of DE: the use of structured population is another possible algorithmic enhancement for DE. A structured population consists in distributing the population's individuals into several subpopulations which evolve independently and interact by exchanging data and information details, contributing to a unique simultaneous evolution. When individuals are exchanged between populations, the process is called migration. Algorithms employing structured populations are also known as distributed algorithms, see Alba and Tomassini (2002). Multiple, small-sized subpopulations

impose a high exploitation pressure within the population, while migration prevents stagnation by injected new, unrelated individuals, effectively refreshing the population's genotype in order to prevent stagnation by modifying the set of possible search moves. If the migration gives precedence to the best solutions, the new search moves promote the detection of promising new search directions and thus allows the DE search structure to be periodically updated. Thus, the migration is supposed to mitigate risk of stagnation of the DE (sub-)populations and to enhance the global algorithmic performance. The employment of structured populations has been applied in literature and has proven to be promising in DE schemes, especially in a large scale optimization since it assists the evolutionary framework in overcoming the curse of dimensionality from which the DE suffers, see Brest and Maučec (2008), Olorunda and Engelbrecht (2007), Noman and Iba (2005), Brest et al. (2008).

Several works have been carried out in the field of distributed DE. In Kwedlo and Bandurski (2006) a distributed DE scheme employing a ring topology (the cores are interconnected in a circle and migrations occur following the ring) has been proposed for the training of a neural network. In Salomon et al. (2005), an example of DE parallelization is given for a medical imaging application. A few famous examples of distributed DE are presented in Zaharie (2002b), Zaharie (2003), and Zaharie and Petcu (2003); in these papers the migration mechanism as well as the algorithmic parameters are adaptively coordinated according to criterion based on genotypical diversity. In Zaharie (2002b), a distributed DE for preserving diversity within the niches is proposed in order to solve multi-modal optimization problems. In Tasoulis et al. (2004), a distributed DE characterized by a ring topology and the migration of individuals with the best performance, to replace random individuals of the neighbor sub-population, has been proposed. An application of the algorithm in Tasoulis et al. (2004) for training of a neural network has been presented in Pavlidis et al. (2005). Following similar logic, Kozlov and Samsonov (2006) proposes a distributed DE where the computational cores are arranged according to a ring topology and, during migration, the best individual of a sub-population replaces the oldest member of the neighboring population. In De Falco et al. (2007a), De Falco et al. (2007b), and De Falco et al. (2007c) a distributed DE has been designed for the image registration problem. In these papers, a computational core acts as a master by collecting the best individuals detected by the various sub-populations running in slave cores. The slave cores are connected in a grid and a migration is arranged among neighbor sub-populations. In Apolloni et al. (2008), a distributed DE which modifies the scheme proposed in Tasoulis et al. (2004) has been presented. In Apolloni et al. (2008), migration is based on a probabilistic criterion depending on five parameters. It is worthwhile mentioning that some parallel implementations of sequential DE (without structured population) are also available in literature, see Nipteni et al. (2006). An investigation of DE parallelization is given in Lampinen (1999).

The combination of structured populations and variable scale factor have been studied in Weber et al. (2011), analyzing the effects on three different variants of DE employing structured populations i.e., those proposed in Tasoulis et al. (2004), Apolloni et al. (2008), and De Falco et al. (2007a), combined to four different schemes varying the value of the scale factor. In the present study, the same algorithms, originally chosen because they are amongst the most popular and most cited modern distributed DE schemes and can thus be considered as state-of-the-art in the field, are modified to employ a DE with an exponential crossover. The study is performed on the same set of various, complex large scale problems, and its goal is to analyze the effect of combining a varying scale factor and the exponential crossover on large-scale problems, and to compare these results to the ones obtained with a binomial crossover. The differences between the results of the present study and those described in Weber et al. (2011) are also presented.

The remainder of this paper is organized in the following way. Section 2 describes the working principles of DE and introduces the notation used throughout this paper. Section 3 gives a short description of the distributed versions of DE presented in Tasoulis et al. (2004), Apolloni et al. (2008), and De Falco et al. (2007a). Section 4 describes the four proposed update schemes for the scale factor. Section 6 shows the experimental setup and numerical results of the present study. Section 7 gives the conclusions of this paper.

2 Differential evolution

In order to clarify the notation used throughout this article we refer to the minimization problem of an objective function $f(x)$, where x is a vector of n design variables in a decision space D. This section gives the description of DE according to its original definition given in Storn and Price (1995). A schematic description of DE highlighting the working principles of the algorithms is given in Fig. 2.

An initial sampling of S_{pop} individuals is performed randomly with a uniform distribution function within the decision space D (for simplicity, the term random will be used instead of random in the remainder of this paper). At each generation, for each individual x_i of the S_{pop}, three individuals x_r, x_s and x_t are randomly extracted from the population. According to the DE logic, a provisional offspring x'_{off} is generated by mutation:

$$x'_{off} = x_t + F(x_r - x_s) \tag{1}$$

where $F \in [0, 1+\epsilon[$ is a scale factor which controls the length of the exploration vector $(x_r - x_s)$ and thus determines how far from point x_i the offspring should be generated. With $F \in [0, 1+\epsilon[$, it is meant here that the scale factor should be a positive value which cannot be much greater than 1 (i.e. ϵ is a small positive value), see Price et al. (2005). While there is no theoretical upper limit for F, effective values are rarely greater than 1.0. The mutation scheme given in Eq. (1) is also known as DE/rand/1. Other variants of the mutation rule have been subsequently proposed in literature, see Qin and Suganthan (2005):

- DE/best/1: $x'_{off} = x_{best} + F(x_s - x_t)$
- DE/current-to-best/1: $x'_{off} = x_i + F(x_{best} - x_i) + F(x_s - x_t)$
- DE/best/2: $x'_{off} = x_{best} + F(x_s - x_t) + F(x_u - x_v)$
- DE/rand/2: $x'_{off} = x_r + F(x_s - x_t) + F(x_u - x_v)$
- DE/current-to-best/2: $x'_{off} = x_i + F(x_{best} - x_i) + F(x_r - x_s) + F(x_u - x_v)$

where x_{best} is the solution with the best performance among individuals of the population, x_u and x_v are two additional randomly selected individuals. It is worthwhile to mention the rotation invariant recombination shown in Price (1999):

- DE/current-to-rand/1 $x_{off} = x_i + K(x_t - x_i) + F'(x_r - x_s)$

where K is the combination coefficient, which, as suggested in Price (1999), should be chosen with a uniform random distribution from [0, 1] and $F' = K \cdot F$. Since this expression includes mutation and crossover (see below), the offspring does not undergo the crossover operation described below.

Fig. 1 Exponential crossover pseudo-code

```
x_off ← x_i
generate j ← 1 + round(n × rand(0,1))
x_off(j) ← x'_off(j)
p ← 0
while rand(0,1) ≤ Cr AND p < n − 1 do
    x_off(1 + (j + p) mod n) ← x'_off(1 + (j + p) mod n)
    p ← p + 1
    generate rand(0,1)
end while
```

Recently, in Price et al. (2005), a new mutation strategy has been defined. This strategy, namely DE/rand/1/either-or and represented in Eq. 2, consists of the following:

$$x'_{off} = \begin{cases} x_t + F\,(x_r - x_s) & \text{if } \; rand\,(0,1) < p_F \\ x_t + K\,(x_r + x_s - 2x_t) & \text{otherwise} \end{cases} \tag{2}$$

where for a given value of F, the parameter K is set equal to $0.5\,(F+1)$.

When the provisional offspring has been generated by mutation, some of the genes of the individual x'_{off} are exchanged with the corresponding gene of x_i in an operation referred to as *crossover*. The most widely used crossover scheme in DE is the binomial crossover, where each gene has a fixed, uniform probability to be exchanged. The variant of DE used in this paper however uses the so-called exponential crossover, where a design variable of the provisional offspring $x'_{off}\,(j)$ is randomly selected and copied into the jth design variable of the solution x_i. This guarantees that parent and offspring have different genotypes. Subsequently, a set of random numbers between 0 and 1 are generated. As long as $rand\,(0,1) \leq Cr$, where the crossover rate Cr is a predetermined parameter, the design variables from the provisional offspring (mutant) are copied into the corresponding positions of the parent x_i. The first time that $rand\,(0,1) > Cr$ the copy process is interrupted. Thus, all the remaining design variables of the offspring are copied from the parent. For the sake of clarity the pseudo-code of the exponential crossover is shown in Fig. 1

The resulting offspring x_{off} is evaluated and, according to a one-to-one spawning strategy, it replaces x_i if and only if $f(x_{off}) < f(x_i)$; otherwise no replacement occurs. It must be remarked that although the replacement indexes are saved one by one during generation, actual replacements occur all at once at the end of the generation. For the sake of clarity, the pseudo-code highlighting working principles of the DE with exponential crossover is shown in Fig. 2.

3 Distributed differential evolution: recently developed algorithms

This section describes three distributed algorithms based on a DE structure recently proposed in literature. The algorithms described in this section are, according to our judgement, representative of the state-of-the-art structured DE algorithms and have been included as distributed structures which can integrate the proposed scale factor update rules. In order to avert misunderstanding regarding the usage of the terms "parallel" or "distributed", the reader must be aware that even though the algorithms described below were originally designed to be run on parallel computers such as clusters or multi-core systems, the subdivision of the population into multiple sub-populations as an algorithmic change from the original definition of the DE has in itself an important impact on the performance of the algorithm. The implementations by the authors of this paper of the algorithms described below were designed to be run on a single processor, by serializing the instructions that would otherwise be run

Fig. 2 DE pseudo-code

```
generate S_pop individuals of the initial population randomly
while budget condition do
    evaluate the fitness values of the population
    for i = 1 : S_pop do
        {** Mutation **}
        select three individuals x_r, x_s, and x_t
        compute x'_off = x_t + F(x_r − x_s)
        {** Crossover **}
        x_off ← x_i
        generate j ← 1 + round(n × rand(0,1))
        x_off(j) ← x'_off(j)
        p ← 0
        while rand(0,1) ≤ Cr AND p < n − 1 do
            x_off(1 + (j + p) mod n) ← x'_off(1 + (j + p) mod n)
            p ← p + 1
            generate rand(0,1)
        end while
        {** Selection **}
        if f(x_off) < f(x_i) then
            save index for replacement x_i = x_off
        end if
    end for
    perform replacements
end while
```

concurrently on a parallel computer. Those serialized algorithms were however actually run on a cluster of ten computers in order to speed-up the process of obtaining multiple, independent trial runs of the same algorithm and therefore statistically significant results. In the descriptions below, we decided to keep the original terminology defined by their respective authors regarding the use of the terms “parallel” or ”distributed”.

3.1 Parallel differential evolution

In Tasoulis et al. (2004), the problem of parallelization for DE schemes has been studied through an experimental analysis and an algorithm, namely Parallel Differential Evolution (PDE) has been proposed.

The original PDE implementation uses the Parallel Virtual Machine (PVM), allowing multiple computers (called *nodes*) to be organized as a cluster and exchange arbitrary messages. PDE is organized around one master node and m sub-populations running each on one node, and organized as a unidirectional ring, as illustrated in Fig. 3. It must be noted that although the logical topology is a ring which does not contain the master node, the actual topology is a star, where all communications (i.e., the migrations of individuals) are passing through the master.

The S_{pop} individuals constituting the populations are distributed over the m sub-populations composing the ring. Each sub-population is composed of $\frac{S_{pop}}{m}$ individuals. Each sub-population runs a regular DE algorithm while the master node coordinates the migration of individuals between sub-populations. On each generation, the sub-population has a given probability ϕ to send a copy of its best individual to its next neighbor sub-population in the ring. When migration occurs, the migrating individual replaces a randomly selected individual belonging to the target sub-population. Figure 4 describes the behavior of both the master node and the sub-populations in more detail.

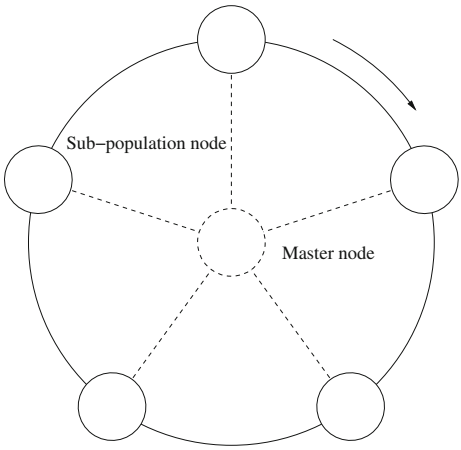


Fig. 3 Unidirectional ring topology in the parallel differential evolution algorithm

```
spawn N sub-populations, each one on a different processor
for each generation do
    receive an individual from each sub-population
    for each received individual do
        if rand(0, 1) < φ then
            send the individual to the next sub-population in the ring
        end if
    end for
    if the stop criterion for the objective function is met then
        send a termination signal to all the sub-populations
    end if
end for
```

(a) At the master node

```
for each generation do
    perform a DE generation
    send a copy of the best individual to the master node
    if a migrated individual has been received then
        replace a random individual, different from the best, by this migrated individual
    end if
    if a termination signal has been received then
        terminate the execution
    end if
end for
```

(b) At each sub-population

Fig. 4 Pseudo-code of the PDE

The DE variant run by each sub-population is the same across all sub-populations. In Tasoulis et al. (2004), six mutation strategies have been compared, namely DE/best/1, DE/rand/1, DE/cur-to-best/1, DE/best/2, DE/rand/2 described in Sect. 2, as well as the trigonometric operator described in Fan and Lampinen (2003). Each strategy is used with different values of the migration constant ϕ and compared over seven test functions whose dimensions vary between 2 and 30. The results in Tasoulis et al. (2004) showed that DE/best/1 is the most efficient mutation strategy and quite stable across different values of ϕ for the low dimensional problems analyzed.

Fig. 5 Pseudo-code of the IBDDE for the sub-population P

```
initialize(P)
while the stopping condition is not met do
    perform a DE generation
    if the last migration was γ generations ago then
        for each of the ρ individuals to send do
            v'^i_g ← φ_s(P)
            send v'^i_g to Q chosen by τ
        end for
    end if{** Asynchronous communication **}
    while individuals are arriving do
        receive v'^i_g from P
        replace an individual chosen from φ_r(Q) by v'^i_g
    end while
end while
```

3.2 Island based distributed differential evolution

In Apolloni et al. (2008) a distributed DE, namely Island Based Distributed Differential Evolution (IBDDE) has been proposed. The IBDDE is a modified version of the PDE described in Sect. 3.1. With the IBDDE the population, having size S_{pop}, is structured in m sub-populations. Thus, each sub-population is composed of $\frac{S_{pop}}{m}$ individuals. The migration policy is then defined as a five-tuple $\mathscr{M} = (\gamma, \rho, \phi_s, \phi_r, \tau)$. $\gamma \in \mathbb{N}$ is the number of generations between two migrations, $\rho \in \mathbb{N}$ is the number of individuals which migrate from a sub-population P during each migration, ϕ_s is the selection function which, applied to a sub-population, returns the migrating individuals v'^i_g, ϕ_r is the replacement function that selects individuals to be replaced by immigrants in the receiving sub-population, and τ is the topological rule, which selects the target sub-population Q. The individuals to be migrated are randomly (uniformly) chosen by the selection function ϕ_s. Incoming individuals from other sub-populations replace randomly chosen local individuals, that is if the former are better, by the replacement function ϕ_r.

Figure 5 describes the algorithm as pseudo-code.

In Apolloni et al. (2008), the experiments have been run with a population size S_{pop} equal to 20. The population is divided into two sub-populations of 10 individuals in one experiment, and into four sub-populations of 5 individuals in a second experiment. The migration parameters are set to $\gamma = 100$, $\rho = 1$, the functions ϕ_s and ϕ_r are defined to randomly select an individual, the topology τ is a unidirectional ring very similar to the logical topology used by PDE (see Sect. 3.1). The mutation strategy for DE is DE/rand/1, and the algorithm is tested on 25 different test functions in 30 and 50 dimensions, for a total of 50 test functions.

3.3 Distributed differential evolution

In De Falco et al. (2007a,b,c), in order to solve some image registration problems a distributed DE (indicated here with DDE) has been proposed. This algorithm differs from PDE and IBDDE by the topology it uses. Instead of a unidirectional ring, DDE uses a locally connected topology, where each node is connected to μ other nodes. Figure 6 represents such a topology, where the nodes are arranged in a mesh folded into a torus.

In De Falco et al. (2007a,b,c), it has been proposed to set $\mu = 4$, i.e. each node (such as the black disc in the Fig. 6) has exactly four nearest neighbors (represented by the four grey discs). In DDE, each node represents one processor running a DE algorithm with a DE/rand/1 mutation strategy on a sub-population. Every M_I generations (the migration interval), each

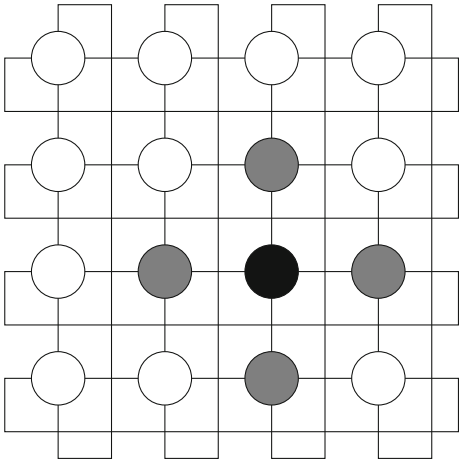


Fig. 6 Torus topology in the distributed differential evolution

```
initialize the sub-population
while the stopping condition is not met do
    perform a DE generation
    if the last migration was M_I generations ago then
        send a copy of the best individual to each neighbor
    end if
    if there are incoming individuals then
        replace the worst S_I × μ individuals by the S_I × μ incoming ones
    end if
end while
```

Fig. 7 Pseudo-code of the DDE algorithm at a sub-population

sub-population is allowed to exchange S_I (the migration rate) individuals with its nearest neighbors. In the experimental setup, each node sends a copy of its best individual to its neighbors. Figure 7 describes the algorithm as pseudo-code.

DDE also makes use of a master node, whose role it is to collect the best solutions found in each sub-population and to present these results to the user.

4 Multiple scale factor variants

As mentioned above, the employment of structured populations greatly improves the performance of DE since each sub-population is supposed to explore the decision space from complementary perspectives, thus enhancing the efficiency in the search for the optimum. With all distributed DE schemes present in literature (see for example the algorithms described in Sect. 3), a unique scale factor is employed for each sub-population and the same scale factor is used throughout the entire optimization process. This paper aims at testing the effect of multiple and variable scale factors on the performance of a distributed DE, in order to propose an enhanced version of distributed DE.

In order to pursue this aim, four simple scale factor variation schemes for distributed DE have been proposed here. Let us consider a generic distributed DE where the S_{pop} individuals of the population are distributed over m sub-populations. The sub-populations evolve

separately, are arranged according to a certain topology (e.g. the ring as in Tasoulis et al. 2004 or the torus as in De Falco et al. 2007a), and interact in some ways (e.g. exchanging individuals).

The scale factors within each sub-population are managed according to one of the following schemes.

4.1 Random initialization of the scale factor: scheme-1

At the beginning of evolution a random scale factor is randomly sampled from a uniform distribution between 0 and 1 for each sub-population. More formally, for the m sub-populations, $F_k = rand(0, 1)$ for $k = 1, 2, \ldots, m$. At each subsequent generation, the scale factors related to each sub-population are kept constant throughout the entire evolution.

4.2 Equally spaced scale factors: scheme-2

The scale factor values are initialized in order to guarantee that the sub-populations work with diverse scale factors. In order to pursue this aim, the scale factors are taken equally spaced within the interval [0, 1], as described in Eq. (3). More specifically, for the m sub-populations,

$$F_k = \frac{(k-1)}{m} + \frac{1}{2m} \tag{3}$$

where $k = 1, 2, \ldots, m$ is the sub-population number. For example, if the distributed system is composed of four sub-populations ($m = 4$), the scale factor values 0.125, 0.375, 0.625, 0.875 are respectively assigned to the sub-populations. The scale factor values are kept constant in each sub-population throughout the entire optimization process.

4.3 Adaptive-randomized update of the scale factor: scheme-3

This scheme, although very simple, is slightly more sophisticated compared to the others considered in this paper. At the beginning of the optimization process, scale factor values (one for each sub-population) are randomly initialized between 0 and 1 from a uniform distribution. Subsequently, the scale factor related to the sub-population which has improved the least over a predetermined number of g_r generations is replaced by another random scale factor sampled from a uniform distribution between [0, 1], see Eq. (4). More formally, every g_r generation, the kth sub-population which satisfies the following criterion is selected:

$$\min_k \left(f_k^g - f_k^{g-g_r} \right) \tag{4}$$

where g is the current generation number, $k = 1, 2, \ldots, m$ is the sub-population index, f_k^g is the fitness value of the best individual of sub-population k at the generation g, and $f_k^{g-g_r}$ is the fitness value of the best individual of sub-population k at the generation $g - g_r$. The scale factor of the selected sub-population is replaced by another random number between 0 and 1 sampled from a uniform distribution.

4.4 Random update of the scale factor: scheme-4

This trivial update scheme can be considered as a distributed version of the dither: for each sub-population, a scale factor value is randomly initialized from a uniform distribution between 0 and 1, $F_k = rand(0, 1)$ for $k = 1, 2, \ldots, m$. At each subsequent generation, a sub-population is randomly selected and its scale factor is replaced by a newly generated random value between 0 and 1, sampled from a uniform distribution.

5 Discussion based around scale factor schemes

All four presented schemes employ multiple scale factors, one for each sub-population. Schemes 1 and 2 assign the scale factor values during the initialization. During the entire evolutionary process, the values are retained. On the contrary, schemes 3 and 4 employ a scale factor update during the evolution.

Scheme-1 simply assumes that employment of multiple scale factors can be beneficial since this allows an enlargement of the set containing the search moves, thus giving a better chance for the algorithm to detect solutions with a high performance.

Scheme-2 makes the same assumption but, in addition, imposes an equal distribution of the scale factor values over the sub-populations. This fact corresponds to an explicit role division among sub-populations: those sub-populations characterized by a low scale factor value are supposed to search in the neighborhood of the solutions while those sub-populations characterized by a high scale factor value are supposed to explore a broader area of the decision space. Obviously, the actual exploration radius of the sub-population search depends on the distribution of sub-population candidate solutions within the decision space. In this sense, the scale factor does not independently control the exploration and exploitation pressure. Nevertheless, the scale factor is an important parameter within a DE scheme because it biases the amplitude of the search radius. Thus, equal spacing of the scale factor values makes sense within the search logic.

In addition to the multiple scale factors, scheme-3 employs a success rule. Several adaptive evolutionary algorithms proposed in literature and, in particular, DE based algorithms (see Qin et al. 2009) also make use of similar ideas. Scheme-3 simply aims at allowing sub-populations which are performing well to continue exploiting their search directions and periodically "refresh" the search logic (by means of the scale factor) of those sub-populations which have become inefficient.

Scheme-4 employs multiple scale factors and a simple update rule based on a random criterion. This scheme, inspired from the dither, aims at providing a constant update to the search logic and relies on the fact that, on average, this mechanism greatly increases the amount of search moves. In addition, this scheme introduces extra randomization within a DE algorithm. As observed in Caponio and Neri (2009), a DE structure can be, for some problems, overwhelmingly deterministic and thus take on a stagnating behavior. The increase of randomization seems an easy and efficient countermeasure to assist DE structures, see Das et al. (2005) and Brest et al. (2006).

6 Results and discussion

In order to prove the viability of the four proposed schemes, the algorithms described in Sect. 3.1 have been considered. Within each of the three distributed DE algorithms the multiple scale factors schemes have been integrated. The test problems listed in Table 1 have been considered in this study.

Table 1 Test problems

Test problem	Function	Decision space
Ackley	$-20 + e + 20 \exp\left(-\frac{0.2}{n}\sqrt{\sum_{i=1}^{n} x_i^2}\right)$	$[-1, 1]^n$
	$-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi \cdot x_i)x_i\right)$	
Alpine	$\sum_{i=1}^{n} \lvert x_i \sin x_i + 0.1x_i \rvert$	$[-10, 10]^n$
Axis-parallel hyper-ellipsoid	$\sum_{i=1}^{n} i x_i^2$	$[-5.12, 5.12]^n$
DeJong	$\lVert x \rVert^2$	$[-5.12, 5.12]^n$
DropWave	$-\frac{1+\cos\left(12\sqrt{\lVert x \rVert^2}\right)}{\frac{1}{2}\lVert x \rVert^2+2}$	$[-5.12, 5.12]^n$
Griewangk	$\frac{\lVert x \rVert^2}{4000} - \prod_{i=0}^{n}\cos\frac{x_i}{\sqrt{i}} + 1$	$[-600, 600]^n$
Michalewicz	$-\sum_{i=1}^{n}\sin x_i \left(\sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right)^{20}$	$[0, \pi]^n$
Pathological	$\sum_{i=1}^{n-1}\left(0.5 + \frac{\sin^2\left(\sqrt{100x_i^2+x_{i+1}^2}-0.5\right)}{1+0.001*\left(x_i^2-2x_i x_{i+1}+x_{i+1}^2\right)^2}\right)$	$[-100, 100]^n$
Rastrigin	$10n + \sum_{i=0}^{n}\left(x_i^2 - 10\cos(2\pi x_i)\right)$	$[-5.12, 5.12]^n$
Rosenbrock valley	$\sum_{i=1}^{n-1}\left(100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2\right)$	$[-2.048, 2.048]^n$
Schwefel	$\sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	$[-500, 500]^n$
Sum of powers	$\sum_{i=1}^{n} \lvert x_i \rvert^{i+1}$	$[-1, 1]^n$
Tirronen	$3\exp\left(-\frac{\lVert x \rVert^2}{10n}\right) - 10\exp\left(-8\lVert x \rVert^2\right)$	$[-10, 5]^n$
	$+\frac{2.5}{n}\sum_{i=1}^{n}\cos\left(5\left(x_i + (1 + i \bmod 2)\cos\left(\lVert x \rVert^2\right)\right)\right)$	

The rotated version of some of the test problems listed in Table 1 have been included in the benchmark set. These rotated problems have been generated through multiplication of the vector of variables by a randomly generated orthogonal rotation matrix. In total, twenty-four test problems have been considered in this study with $n = 500$. In order to obtain statistically significant results, each algorithm considered in this paper has been run 50 times (50 independent runs) for 500,000 fitness evaluations.

One must make an important remark concerning the difference in the behaviours of the binomial crossover compared to the exponential crossover. The binomial crossover will exchange on average $n \times CR$ variables between the parent and the provisional offspring, which for $n = 500$ and $CR = 0.3$ (the values used in Weber et al. 2011), is equal to 150. The number of exchanged variables in the case of the exponential crossover however does not depend on n, but is based on a geometric distribution of the form $Cr^k(1-Cr)$. When one sets $CR = 0.9$ as is the case in the experiments described below, the mean number of variables which are exchanged is $Cr/(1 - Cr) = 9$, which is one order of magnitude smaller than the average number of variables exchanged when using the binomial crossover. Moreover, the

order of magnitude of the probability of exchanging 150 variables or more is 10^{-7}, which is much less than once in the whole evolutionary process.

6.1 Parallel differential evolution: numerical results

The PDE described in Sect. 3.1 has been run and the fours schemes presented in Sect. 4 have been integrated within the PDE. The PDE is indicated by PDE_1, PDE_2, PDE_3, and PDE_4 and integrates the schemes 1,2,3, and 4, respectively. All the algorithms in this subsection have been run with a population size of $S_{pop} = 200$ individuals, divided into $m = 5$ subpopulations, each containing 40 individuals. Despite Tasoulis et al. (2004) showing better performance for the DE/best/1 mutation strategy in 30 and 50 dimensions, it has proven excessively exploitative and has led to premature convergence of the solutions when used on higher dimension problems in our preliminary tests. We have carried out an analysis on mutation strategies, leading to the choice of DE/rand/1 and, with reference to Sect. 3.1, setting the migration constant to $\phi = 0.2$ as suggested in Weber et al. (2009). These settings proved to be the best choices in terms of algorithmic performance and have, thus, been chosen for the experiments of all the algorithms considered in this subsection. Regarding scale factor F and crossover rate CR of the PDE, the values of 0.5 and 0.9 have been chosen in accordance with suggestions given in Herrera et al. (2010). The crossover rate $CR = 0.9$ has also been employed for all the PDE variants under investigation. In PDE_3, the parameter g_r has been set equal to 30 on the basis of our preliminary tests.

Choice of the population size is worthwile commenting on. Although in Storn and Price (1997) it is suggested that the DE population size be set equal to about ten times the dimensionality of the problem, this indication is not confirmed by a recent study in Neri and Tirronen (2008) where it is shown that a population size lower than the dimensionality of the problem can be optimal in many cases.

Table 2 shows the average of the final results detected by each algorithm $\pm$ the standard deviation values.

To prove the statistical significance of the results, the Wilcoxon Rank-sum test has been applied according to the description given in Wilcoxon (1945) for a confidence level of 0.95. Table 3 shows results of the test. More specifically, Table 3 shows a ranking of the algorithms based on the statistical significance of the results. If one algorithm significantly outperforms another algorithm, they appear in separate cells (over the same row). If, on the contrary, there is no statistical significance, the algorithms have the same ranking (they appear in the same cell).

In order to carry out a numerical comparison of the convergence speed performance and algorithmic robustness for each test problem, the average final fitness value returned by the best performing algorithm G has been considered. Subsequently, the average fitness value at the beginning of the optimization process J has also been computed. The threshold value $THR = J - 0.95(J - G)$ has then been calculated. The value THR represents 95% of the decay occurring in the fitness value of the algorithm with the best performance. If an algorithm succeeds during a certain run in reaching the value THR, the run is said to be successful. For each test problem, the average amount of fitness evaluations $\bar{ne}$ required, for each algorithm, to reach THR has been computed. Subsequently, the Q-test (Q stands for Quality) described in Feoktistov (2006) has been applied. For each test problem and each algorithm, the Q measure is computed as:

$$Q = \frac{\bar{ne}}{R} \tag{5}$$

Table 2 Average final fitness values ± standard deviations for PDE and its multiple scale factors variants

	PDE	PDE_1	PDE_2	PDE_3	PDE_4
Ackley	$\mathbf{3.09e-02 \pm 2.23e-03}$	$4.11e-02 \pm 9.43e-03$	$3.84e-02 \pm 3.52e-03$	$4.31e-02 \pm 6.89e-03$	$4.05e-02 \pm 3.42e-03$
Alpine	$3.51e+01 \pm 4.85e+00$	$3.28e+01 \pm 5.39e+00$	$3.03e+01 \pm 3.72e+00$	$2.82e+01 \pm 4.33e+00$	$\mathbf{2.55e+01 \pm 3.05e+00}$
Ax.-par. hyp.-ell.	$\mathbf{1.83e+02 \pm 2.08e+01}$	$5.12e+02 \pm 5.24e+02$	$3.43e+02 \pm 5.50e+01$	$4.03e+02 \pm 1.05e+02$	$3.60e+02 \pm 6.64e+01$
DeJong	$\mathbf{8.32e-01 \pm 9.19e-02}$	$1.72e+00 \pm 8.74e-01$	$1.53e+00 \pm 2.05e-01$	$1.63e+00 \pm 3.76e-01$	$1.56e+00 \pm 2.44e-01$
DropWave	$\mathbf{-2.22e-03 \pm 1.62e-04}$	$-1.92e-03 \pm 1.47e-04$	$-1.94e-03 \pm 1.15e-04$	$-1.94e-03 \pm 1.29e-04$	$-1.91e-03 \pm 1.27e-04$
Griewangk	$\mathbf{5.03e+02 \pm 4.35e-01}$	$5.06e+02 \pm 2.33e+00$	$5.05e+02 \pm 6.72e-01$	$5.06e+02 \pm 1.41e+00$	$5.05e+02 \pm 7.99e-01$
Michalewicz	$-3.96e+02 \pm 3.50e+00$	$-3.98e+02 \pm 7.44e+00$	$-3.98e+02 \pm 3.99e+00$	$-4.00e+02 \pm 4.63e+00$	$\mathbf{-4.02e+02 \pm 3.56e+00}$
Pathological	$-2.45e+02 \pm 3.22e+00$	$-2.53e+02 \pm 1.09e+01$	$-2.53e+02 \pm 4.32e+00$	$\mathbf{-2.54e+02 \pm 6.34e+00}$	$-2.53e+02 \pm 5.16e+00$
Rastrigin	$9.59e+02 \pm 4.97e+01$	$8.74e+02 \pm 7.08e+01$	$8.71e+02 \pm 5.29e+01$	$8.23e+02 \pm 4.57e+01$	$\mathbf{8.18e+02 \pm 3.71e+01}$
Rosenbrock	$\mathbf{6.70e+02 \pm 7.55e+01}$	$8.98e+02 \pm 1.71e+02$	$8.61e+02 \pm 8.90e+01$	$9.06e+02 \pm 1.25e+02$	$8.62e+02 \pm 1.04e+02$
Schwefel	$-1.69e+05 \pm 1.86e+03$	$-1.82e+05 \pm 5.23e+03$	$-1.81e+05 \pm 1.87e+03$	$-1.82e+05 \pm 2.98e+03$	$\mathbf{-1.82e+05 \pm 1.85e+03}$
Sum of powers	$8.34e-14 \pm 5.84e-13$	$1.84e-15 \pm 1.03e-14$	$\mathbf{1.91e-17 \pm 7.14e-17}$	$8.39e-17 \pm 2.82e-16$	$4.69e-17 \pm 1.90e-16$
Tirronen	$-8.17e-01 \pm 1.79e-01$	$-8.50e-01 \pm 1.59e-01$	$-8.38e-01 \pm 1.60e-01$	$-8.71e-01 \pm 1.61e-01$	$\mathbf{-8.89e-01 \pm 1.50e-01}$
Rt. Ackley	$\mathbf{1.26e-01 \pm 2.48e-02}$	$1.85e-01 \pm 5.89e-02$	$1.67e-01 \pm 3.60e-02$	$1.72e-01 \pm 3.66e-02$	$1.59e-01 \pm 3.11e-02$
Rt. Alpine	$9.27e+01 \pm 1.10e+01$	$9.13e+01 \pm 1.05e+01$	$8.80e+01 \pm 1.03e+01$	$8.72e+01 \pm 8.41e+00$	$\mathbf{7.97e+01 \pm 7.87e+00}$
Rt. Ax.-par. hyp.-ell.	$\mathbf{2.02e+03 \pm 3.22e+02}$	$3.10e+03 \pm 9.54e+02$	$2.88e+03 \pm 5.05e+02$	$3.05e+03 \pm 5.52e+02$	$2.78e+03 \pm 4.46e+02$
Rt. Griewangk	$\mathbf{5.17e+02 \pm 2.69e+00}$	$5.25e+02 \pm 6.88e+00$	$5.25e+02 \pm 4.44e+00$	$5.26e+02 \pm 4.79e+00$	$5.25e+02 \pm 3.95e+00$
Rt. Michalewicz	$-2.18e+02 \pm 3.38e+00$	$-2.26e+02 \pm 5.82e+00$	$-2.26e+02 \pm 5.14e+00$	$-2.27e+02 \pm 3.57e+00$	$\mathbf{-2.29e+02 \pm 3.97e+00}$
Rt. Pathological	$-1.76e+02 \pm 5.49e+00$	$-1.80e+02 \pm 1.09e+01$	$\mathbf{-1.86e+02 \pm 3.93e+00}$	$-1.83e+02 \pm 5.26e+00$	$-1.83e+02 \pm 4.82e+00$
Rt. Rastrigin	$1.52e+03 \pm 8.65e+01$	$1.46e+03 \pm 7.78e+01$	$1.46e+03 \pm 6.34e+01$	$1.43e+03 \pm 7.72e+01$	$\mathbf{1.40e+03 \pm 7.37e+01}$
Rt. Rosenbrock	$\mathbf{1.04e+03 \pm 1.22e+02}$	$1.32e+03 \pm 2.33e+02$	$1.27e+03 \pm 1.61e+02$	$1.31e+03 \pm 1.61e+02$	$1.26e+03 \pm 1.22e+02$
Rt. Schwefel	$-1.48e+05 \pm 2.72e+03$	$-1.56e+05 \pm 3.86e+03$	$-1.56e+05 \pm 2.67e+03$	$-1.56e+05 \pm 2.90e+03$	$\mathbf{-1.58e+05 \pm 3.23e+03}$
Rt. Sum of powers	$3.57e-07 \pm 2.43e-06$	$1.03e-07 \pm 7.16e-07$	$6.58e-13 \pm 4.58e-12$	$9.57e-09 \pm 4.72e-08$	$\mathbf{2.50e-13 \pm 1.09e-12}$
Rt. Tirronen	$-8.29e-01 \pm 1.42e-01$	$-8.27e-01 \pm 1.48e-01$	$-8.73e-01 \pm 1.19e-01$	$-8.81e-01 \pm 1.48e-01$	$\mathbf{-8.81e-01 \pm 1.54e-01}$

Table 3 Results of the Wilcoxon rank-sum test for the PDE and its variants

	1	2	3	4	5
Ackley	PDE	PDE_1 PDE_2 PDE_4			PDE_3
Alpine	PDE_4	PDE_3	PDE_2	PDE PDE_1	
Ax.-par. hyp.-ell.	PDE	PDE_2	PDE_1 PDE_3 PDE_4		
DeJong	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
DropWave	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
Griewangk	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
Michalewicz	PDE_4	PDE_1 PDE_2 PDE_3			PDE
Pathological	PDE_1 PDE_2 PDE_3 PDE_4				PDE
Rastrigin	PDE_3 PDE_4		PDE_1 PDE_2		PDE
Rosenbrock	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
Schwefel	PDE_1 PDE_3 PDE_4			PDE_2	PDE
Sum of powers	PDE_4	PDE_1 PDE_2 PDE_3			PDE
Tirronen	PDE_4	PDE PDE_1 PDE_2 PDE_3			
Rt. Ackley	PDE	PDE_4	PDE_1 PDE_2 PDE_3		
Rt. Alpine	PDE_4	PDE_2 PDE_3		PDE PDE_1	
Rt. Ax.-par. hyp.-ell.	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
Rt. Griewangk	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
Rt. Michalewicz	PDE_4	PDE_1 PDE_2 PDE_3			PDE
Rt. Pathological	PDE_2	PDE_1 PDE_3 PDE_4			PDE
Rt. Rastrigin	PDE_3 PDE_4		PDE_1 PDE_2		PDE
Rt. Rosenbrock	PDE	PDE_1 PDE_2 PDE_3 PDE_4			
Rt. Schwefel	PDE_4	PDE_1 PDE_2 PDE_3			PDE
Rt. Sum of powers	PDE_2 PDE_3		PDE PDE_1 PDE_4		
Rt. Tirronen	PDE_4	PDE PDE_1 PDE_2 PDE_3			

where the robustness R is the percentage of successful runs. It is clear that, for each test problem, the smallest value equals the best performance in terms of convergence speed. The value "∞" means that $R = 0$, i.e., the algorithm never reached the *THR*.

Table 4 shows the Q values for PDE and its variants. The best results are highlighted in bold face.

Some performance trends (averaged over 50 independent runs) of the PDE (solid line) and its multiple scale factors variants (dashed and dotted lines) are displayed in Fig. 8.

Numerical results show that in nineteen cases out of the twenty-four considered, the standard PDE or the Scheme 4 significantly outperform the other schemes, see Tables 2 and 3. The standard PDE is ranked first in ten cases, while the Scheme 4 is ranked first in twelve cases. It is worth noting that in the cases when Scheme 4 is ranked first, the standard PDE is systematically ranked last, while in the cases where the standard PDE is ranked first, the Scheme 4 is ranked second (with only one exception). Table 3 also shows that in twenty cases, three or more algorithms are ranked together, meaning that there is no significant difference in performance between them. Table 4 shows that the standard PDE exhibits one ∞ value

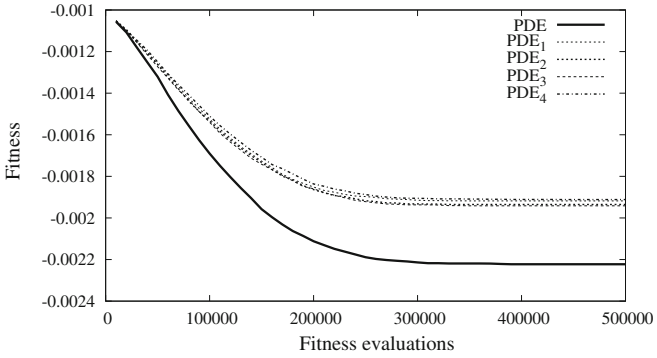
Table 4 Results of the Q-test for the PDE and its variants

	PDE	PDE_1	PDE_2	PDE_3	PDE_4
Ackley	2.89*e*+03	2.92*e*+03	**2.89*e*+03**	2.96*e*+03	3.01*e*+03
Alpine	3.19*e*+03	3.16*e*+03	3.14*e*+03	**3.08*e*+03**	3.10*e*+03
Ax.-par. hyp.-ell.	**1.57*e*+03**	1.61*e*+03	1.57*e*+03	1.61*e*+03	1.66*e*+03
DeJong	1.63*e*+03	1.63*e*+03	**1.62*e*+03**	1.64*e*+03	1.71*e*+03
DropWave	**2.64*e*+03**	2.52*e*+04	1.32*e*+05	4.13*e*+04	5.65*e*+04
Griewangk	1.63*e*+03	**1.61*e*+03**	1.61*e*+03	1.64*e*+03	1.70*e*+03
Michalewicz	4.27*e*+03	4.34*e*+03	4.24*e*+03	4.13*e*+03	**4.09*e*+03**
Pathological	5.17*e*+03	5.28*e*+03	4.61*e*+03	4.58*e*+03	**4.51*e*+03**
Rastrigin	3.70*e*+03	3.35*e*+03	3.41*e*+03	**3.29*e*+03**	3.38*e*+03
Rosenbrock	1.21*e*+03	1.22*e*+03	**1.21*e*+03**	1.23*e*+03	1.27*e*+03
Schwefel	∞	4.06*e*+03	3.91*e*+03	**3.79*e*+03**	3.91*e*+03
Sum of powers	2.00*e*+02	1.98*e*+02	2.06*e*+02	**1.89*e*+02**	1.96*e*+02
Tirronen	5.58*e*+03	4.69*e*+03	5.47*e*+03	4.73*e*+03	**4.09*e*+03**
Rt. Ackley	**3.31*e*+03**	4.10*e*+03	3.76*e*+03	3.79*e*+03	3.67*e*+03
Rt. Alpine	3.32*e*+03	3.34*e*+03	3.27*e*+03	3.25*e*+03	**3.18*e*+03**
Rt. Ax.-par. hyp.-ell.	**1.60*e*+03**	1.67*e*+03	1.64*e*+03	1.67*e*+03	1.69*e*+03
Rt. Griewangk	**1.61*e*+03**	1.65*e*+03	1.65*e*+03	1.68*e*+03	1.72*e*+03
Rt. Michalewicz	8.70*e*+03	4.37*e*+03	4.37*e*+03	4.01*e*+03	**3.93*e*+03**
Rt. Pathological	8.95*e*+03	7.15*e*+03	**4.39*e*+03**	4.87*e*+03	4.99*e*+03
Rt. Rastrigin	3.68*e*+03	3.52*e*+03	3.53*e*+03	3.41*e*+03	**3.41*e*+03**
Rt. Rosenbrock	**1.18*e*+03**	1.24*e*+03	1.21*e*+03	1.23*e*+03	1.25*e*+03
Rt. Schwefel	2.55*e*+04	4.42*e*+03	3.88*e*+03	3.91*e*+03	**3.79*e*+03**
Rt. Sum of powers	4.32*e*+00	4.12*e*+00	**4.00*e*+00**	4.24*e*+00	4.00*e*+00
Rt. Tirronen	5.22*e*+03	4.78*e*+03	4.10*e*+03	**3.56*e*+03**	3.90*e*+03

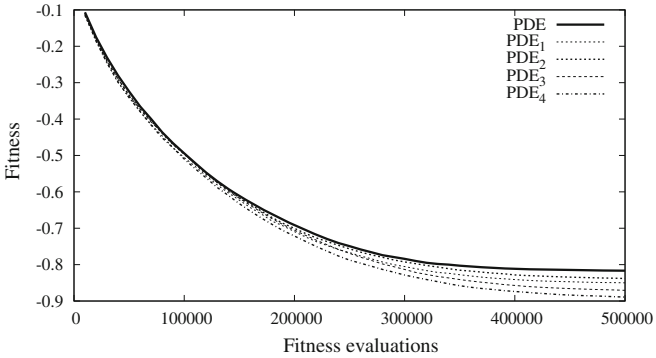
(for the Schwefel function), implying that it is less robust than the other algorithms. The best results in that table (highlighted with a boldface font) are spread over the five algorithms, confirming the fact that there is not one clear winner in this comparison.

Figure 8b and c are illustrations of the cases where Scheme 4 obtains marginally better performances than the original algorithm, while Fig. 8a shows a situation where the original PDE algorithm outperformes all its variants by a large margin.

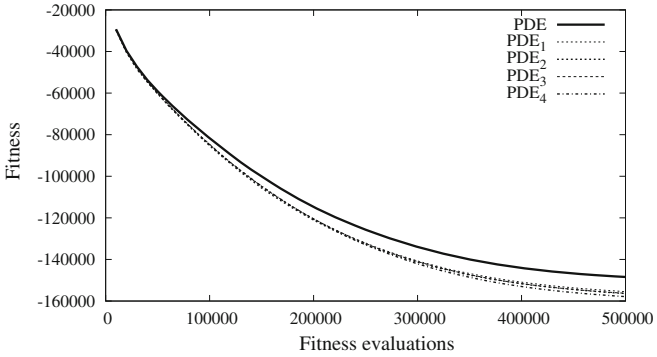
The results above are quite different from those presented in Weber et al. (2011), which concluded that the original PDE with the binomial crossover is exhibiting in general inferior performance than the variants (in that paper, the original PDE was ranked last in seventeen cases out of twenty-four). Moreover, when using the algorithms' average ranks over all the twenty-four test functions as an indicator of their performance, in the previous study, Scheme 3 was considered to be the one with the best overall performance, while in the present study, Scheme 4 exhibits the best overall performance. Nonetheless, the results obtained in the current study tend to indicate that multiple F values do not significantly improve or deteriorate the performances of the PDE with exponential crossover. According to our interpretation, the difference arises from the effect of the binomial crossover compared to the exponential crossover: the latter exchanges very few variables between the parent and the provisional offspring, meaning that the offsring does not step very far from its parent. This



(a) Dropwave



(b) Tirronen



(c) Rotated Schwefel

Fig. 8 Performance trends of PDE and its variants

limits very much DE's aptitude at exploring the problem space, which is not counterbalanced by the use of multiple values of F.

6.2 Island based distributed differential evolution: numerical results

The IBDDE has been run with populations of 200 individuals divided into 5 sub-populations of 40 individuals each, depending on the dimensionality of the test problems. The crossover

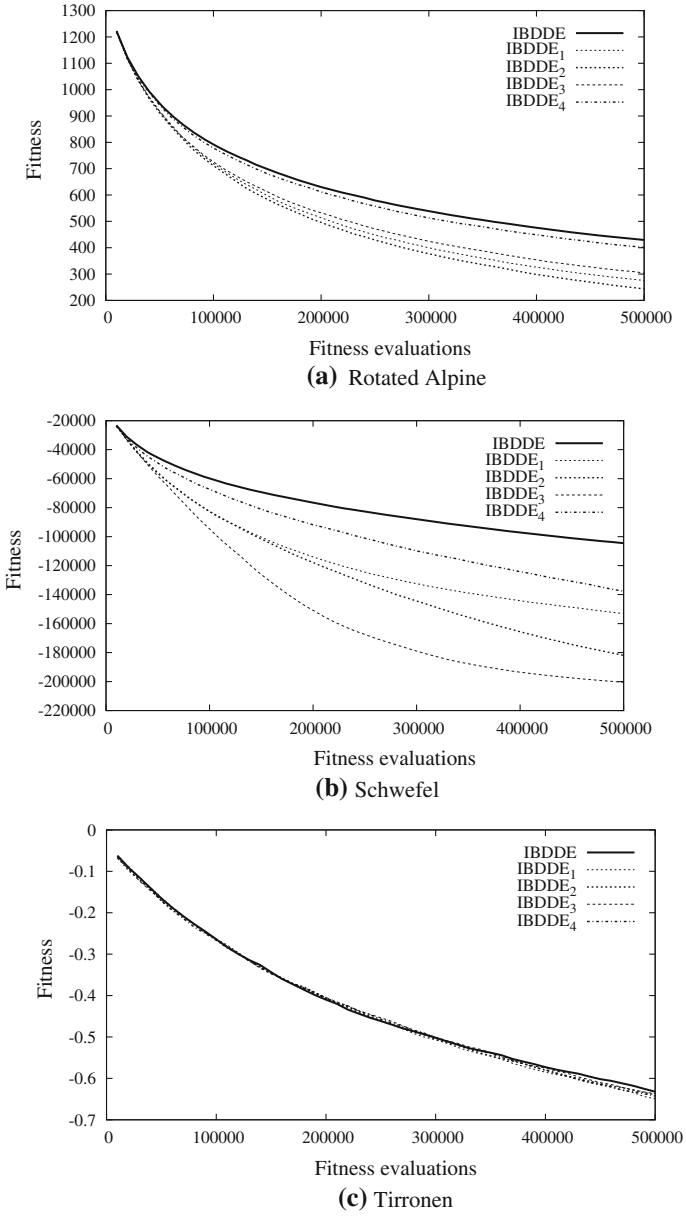


Fig. 9 Performance trends of IBDDE and its variants

rate for all the IBDDE based algorithms has been chosen equal to 0.9 and the scale factor of the IBDDE has been selected equal to 0.5 in accordance to the settings in Herrera et al. (2010). The other parameters have been chosen according to the values in Apolloni et al. (2008): the sub-populations exchange one individual ($\rho = 1$) every 100 generations ($\gamma = 100$). ϕ_s and ϕ_r have been defined so as to randomly select an individual by means of a uniform distribution, and τ has been set to a unidirectional ring.

Analogous to the PDE, the IBDDE integrating the schemes 1, 2, 3, and 4 are indicated as IBDDE_1, IBDDE_2, IBDDE_3, and IBDDE_4. In IBDDE_3, g_r has been set equal to 30. Table 5 lists final average fitness values over 50 independent runs and respective standard deviation values. Tables 6 and 7 show the results of the Wilcoxon rank-sum test and Q-test respectively. The best results are highlighted in bold face.

Some performance trends (averaged over 50 independent runs) of the IBDDE (solid line) and its multiple scale factors variants (dashed and dotted lines) are displayed in Fig. 9.

Numerical results show that, for IBDDE structures, employment of multiple scale factors can be highly beneficial. The standard IBDDE is outperformed by all its variants in ninteen cases out of the twenty-four considered in this study, and it is outperformed by at least one of its variants in twenty-two cases (see Table 6). While the employment of a multiple scale factors scheme does not lead to benefits to absolutely all cases (the original IBDDE is ranked first in two cases), it is never detrimental (when IBDDE is randked first, it is never the only algorithm in that position). It can be seen that Scheme 2 is ranked first in twenty-one case, and second-best in the three other cases. When computing the average rank of the algorithms, Scheme 3 is close second to Scheme 2. However when considering the results of the Q-test (see Table 7), the absence of ∞ signs in the results of Scheme 3 (compared to the presence of three such symbols for Scheme 2), indicate that while the former is presentes a slightly lower performance, it can be considered as a much more stable algorithm. Scheme 1 is stable as well but of lower performance than the two former ones. Finally, the original IBDDE and Scheme 4 exhibit both poorer performance and lower stability.

For illustration purposes, Fig. 9a illustrates the case where Scheme 2 obtains the best results, while Fig. 9b shows Scheme 3 outperforming all the other variants. Finally, Fig. 9c is a counterexample, where the original algorithm and all its variants obtain very similar results, and where the benefits of multiple values of F are not obvious.

These results are very similar to those obtained in Weber et al. (2011), where Schemes 2 and 3 were exhibiting the best performance, with a slight favor for Scheme 3 when considering the stability of the algorithms. This means that, while multiple values of F across the different subpopulations significantly improve the performance of the IBDDE algorithm, in contrast with the results obtained about PDE, it is not possible to tell whether static or dynamic values of the scale factor are preferable. The reason why multiple values of the scale factor are beneficial to IBDDE but not to PDE may be explained by the fact that in IBDDE a random individual is migrated between subpopulations, on the contrary to PDE where the best individual is migrated. The latter behavior focuses the algorithm's attention around current best solutions, and this fact is exacerbated by the highly exploitative character of the exponential crossover. Such a focus point does not exist in IBDDE, where the introduction in a subpopulation, by means of the migration, of a new individual, compensates the exploitative character of the exponential crossover.

6.3 Distributed differential evolution: numerical results

The DDE has been run with a population of 200 individuals arranged according to the torus topology represented in Fig. 6. Following the suggestions in De Falco et al. (2007b), the mesh is composed of 4×4 sub-populations ($\mu = 4$) containing 12 or 13 individuals (8 sub-populations are composed of 12 candidate solutions and the other 8 sub-populations of 13 solutions). Migration occurred in each sub-population with only its best individual ($S_I = 1$) every $M_I = 5$ generations. The crossover rate for all the DDE based algorithms has been chosen equal to 0.9 and the scale factor of the DDE has been selected equal to 0.5 in accordance to the settings in Herrera et al. (2010).

Table 5 Average final fitness values ± standard deviations for IBDDE and its multiple scale factors variants

	IBDDE	$IBDDE_1$	$IBDDE_2$	$IBDDE_3$	$IBDDE_4$
Ackley	$2.33e-01 \pm 4.44e-03$	$6.30e-02 \pm 8.57e-02$	$\mathbf{2.89e-02 \pm 1.36e-03}$	$3.50e-02 \pm 6.14e-03$	$1.03e-01 \pm 8.25e-03$
Alpine	$3.19e+02 \pm 4.70e+00$	$5.63e+01 \pm 8.09e+01$	$\mathbf{1.26e+01 \pm 2.27e+00}$	$1.80e+01 \pm 9.28e+00$	$1.70e+02 \pm 1.42e+01$
Ax.-par. hyp.-ell.	$5.04e+03 \pm 1.72e+02$	$2.12e+03 \pm 4.32e+03$	$\mathbf{2.01e+02 \pm 1.68e+01}$	$3.42e+02 \pm 8.71e+01$	$1.64e+03 \pm 2.35e+02$
DeJong	$2.17e+01 \pm 6.67e-01$	$4.82e+00 \pm 7.25e+00$	$\mathbf{8.44e-01 \pm 7.09e-02}$	$1.49e+00 \pm 5.14e-01$	$7.29e+00 \pm 8.31e-01$
DropWave	$-1.36e-03 \pm 2.51e-05$	$-1.45e-03 \pm 6.18e-05$	$\mathbf{-1.47e-03 \pm 5.17e-05}$	$-1.42e-03 \pm 5.03e-05$	$-1.37e-03 \pm 3.20e-05$
Griewangk	$5.74e+02 \pm 3.04e+00$	$5.32e+02 \pm 5.70e+01$	$\mathbf{5.03e+02 \pm 2.17e-01}$	$5.05e+02 \pm 1.86e+00$	$5.25e+02 \pm 2.47e+00$
Michalewicz	$-2.12e+02 \pm 1.71e+00$	$-2.30e+02 \pm 2.48e+01$	$-2.23e+02 \pm 2.83e+00$	$\mathbf{-2.68e+02 \pm 7.59e+01}$	$-2.15e+02 \pm 2.21e+00$
Pathological	$-8.85e+01 \pm 2.03e+00$	$\mathbf{-1.13e+02 \pm 3.26e+01}$	$-1.04e+02 \pm 1.64e+00$	$-1.09e+02 \pm 3.54e+01$	$-9.25e+01 \pm 1.79e+00$
Rastrigin	$3.19e+03 \pm 3.63e+01$	$1.75e+03 \pm 9.09e+02$	$9.11e+02 \pm 5.83e+01$	$\mathbf{4.80e+02 \pm 2.07e+02}$	$2.34e+03 \pm 1.09e+02$
Rosenbrock	$1.66e+03 \pm 5.17e+01$	$8.37e+02 \pm 5.40e+02$	$\mathbf{5.56e+02 \pm 1.24e+01}$	$5.96e+02 \pm 2.95e+01$	$8.82e+02 \pm 5.66e+01$
Schwefel	$-1.05e+05 \pm 9.60e+02$	$-1.53e+05 \pm 3.62e+04$	$-1.82e+05 \pm 2.34e+03$	$\mathbf{-2.00e+05 \pm 4.07e+03}$	$-1.38e+05 \pm 3.65e+03$
Sum of powers	$7.70e-09 \pm 6.51e-09$	$3.06e-08 \pm 9.79e-08$	$\mathbf{3.46e-11 \pm 1.21e-10}$	$1.41e-10 \pm 3.73e-10$	$5.35e-10 \pm 6.46e-10$
Tirronen	$-6.32e-01 \pm 2.10e-02$	$\mathbf{-6.49e-01 \pm 3.94e-02}$	$-6.39e-01 \pm 2.79e-02$	$-6.37e-01 \pm 2.67e-02$	$-6.43e-01 \pm 2.69e-02$
Rt. Ackley	$7.22e-01 \pm 2.56e-02$	$1.85e-01 \pm 2.29e-01$	$1.34e-01 \pm 1.09e-02$	$\mathbf{1.23e-01 \pm 1.93e-02}$	$3.56e-01 \pm 3.43e-02$
Rt. Alpine	$4.30e+02 \pm 7.62e+00$	$2.76e+02 \pm 8.27e+01$	$\mathbf{2.43e+02 \pm 1.60e+01}$	$3.04e+02 \pm 5.03e+01$	$4.01e+02 \pm 9.41e+00$
Rt. Ax.-par. hyp.-ell.	$2.85e+04 \pm 1.12e+03$	$5.89e+03 \pm 6.16e+03$	$4.39e+03 \pm 5.55e+02$	$\mathbf{3.66e+03 \pm 9.88e+02}$	$1.41e+04 \pm 1.52e+03$
Rt. Griewangk	$7.47e+02 \pm 1.12e+01$	$6.00e+02 \pm 1.60e+02$	$5.24e+02 \pm 2.82e+00$	$\mathbf{5.24e+02 \pm 5.15e+00}$	$5.94e+02 \pm 1.21e+01$
Rt. Michalewicz	$-1.25e+02 \pm 1.38e+00$	$-1.29e+02 \pm 3.02e+00$	$\mathbf{-1.30e+02 \pm 2.12e+00}$	$-1.27e+02 \pm 2.34e+00$	$-1.25e+02 \pm 1.48e+00$
Rt. Pathological	$-7.37e+01 \pm 1.21e+00$	$-7.42e+01 \pm 1.51e+00$	$-7.40e+01 \pm 1.51e+00$	$\mathbf{-7.43e+01 \pm 1.68e+00}$	$-7.39e+01 \pm 1.34e+00$
Rt. Rastrigin	$3.76e+03 \pm 4.57e+01$	$\mathbf{2.90e+03 \pm 6.50e+02}$	$3.12e+03 \pm 6.03e+01$	$3.25e+03 \pm 3.24e+02$	$3.68e+03 \pm 4.97e+01$
Rt. Rosenbrock	$4.67e+03 \pm 1.73e+02$	$1.38e+03 \pm 9.55e+02$	$1.24e+03 \pm 1.31e+02$	$\mathbf{1.15e+03 \pm 1.44e+02}$	$2.95e+03 \pm 2.48e+02$
Rt. Schwefel	$-9.38e+04 \pm 1.44e+03$	$\mathbf{-1.07e+05 \pm 1.29e+04}$	$-1.06e+05 \pm 1.78e+03$	$-1.01e+05 \pm 4.90e+03$	$-9.59e+04 \pm 1.14e+03$
Rt. Sum of powers	$6.47e-06 \pm 5.02e-06$	$2.78e-06 \pm 4.89e-06$	$\mathbf{2.24e-06 \pm 2.65e-06}$	$2.61e-06 \pm 2.90e-06$	$4.41e-06 \pm 4.17e-06$
Rt. Tirronen	$-6.38e-01 \pm 2.58e-02$	$-6.37e-01 \pm 2.56e-02$	$\mathbf{-6.45e-01 \pm 2.49e-02}$	$-6.38e-01 \pm 2.38e-02$	$-6.35e-01 \pm 2.70e-02$

Table 6 Results of the Wilcoxon rank-sum test for the IBDDE and its variants

	1	2	3	4	5
Ackley	$IBDDE_2$	$IBDDE_1$ $IBDDE_3$		$IBDDE_4$	IBDDE
Alpine	$IBDDE_2$	$IBDDE_3$	$IBDDE_1$	$IBDDE_4$	IBDDE
Ax.-par. hyp.-ell.	$IBDDE_2$	$IBDDE_3$	$IBDDE_4$	$IBDDE_1$	IBDDE
DeJong	$IBDDE_2$	$IBDDE_1$ $IBDDE_3$		$IBDDE_4$	IBDDE
DropWave	$IBDDE_1$ $IBDDE_2$		$IBDDE_3$	$IBDDE_4$	IBDDE
Griewangk	$IBDDE_2$	$IBDDE_3$	$IBDDE_4$	$IBDDE_1$	IBDDE
Michalewicz	$IBDDE_1$ $IBDDE_2$ $IBDDE_3$			$IBDDE_4$	IBDDE
Pathological	$IBDDE_1$ $IBDDE_2$ $IBDDE_3$			$IBDDE_4$	IBDDE
Rastrigin	$IBDDE_3$	$IBDDE_2$	$IBDDE_1$	$IBDDE_4$	IBDDE
Rosenbrock	$IBDDE_2$	$IBDDE_1$ $IBDDE_3$		$IBDDE_4$	IBDDE
Schwefel	$IBDDE_3$	$IBDDE_2$	$IBDDE_1$ $IBDDE_4$		IBDDE
Sum of powers	$IBDDE_2$	$IBDDE_3$	$IBDDE_4$	IBDDE	$IBDDE_1$
Tirronen	$IBDDE_1$	IBDDE $IBDDE_2$ $IBDDE_3$ $IBDDE_4$			
Rt. Ackley	$IBDDE_1$ $IBDDE_2$ $IBDDE_3$			$IBDDE_4$	IBDDE
Rt. Alpine	$IBDDE_2$	$IBDDE_1$	$IBDDE_3$	$IBDDE_4$	IBDDE
Rt. Ax.-par. hyp.-ell.	$IBDDE_2$	$IBDDE_1$ $IBDDE_3$		$IBDDE_4$	IBDDE
Rt. Griewangk	$IBDDE_2$ $IBDDE_3$		$IBDDE_4$	$IBDDE_1$	IBDDE
Rt. Michalewicz	$IBDDE_1$ $IBDDE_2$		$IBDDE_3$	IBDDE $IBDDE_4$	
Rt. Pathological	IBDDE $IBDDE_1$ $IBDDE_2$ $IBDDE_3$ $IBDDE_4$				
Rt. Rastrigin	$IBDDE_1$ $IBDDE_2$		$IBDDE_3$	$IBDDE_4$	IBDDE
Rt. Rosenbrock	$IBDDE_2$	$IBDDE_1$ $IBDDE_3$		$IBDDE_4$	IBDDE
Rt. Schwefel	$IBDDE_1$ $IBDDE_2$ $IBDDE_3$			$IBDDE_4$	IBDDE
Rt. Sum of powers	$IBDDE_1$ $IBDDE_2$ $IBDDE_3$			$IBDDE_4$	IBDDE
Rt. Tirronen	IBDDE $IBDDE_1$ $IBDDE_2$ $IBDDE_3$ $IBDDE_4$				

Table 7 Results of the Q-test for the IBDDE and its variants

	IBDDE	$IBDDE_1$	$IBDDE_2$	$IBDDE_3$	$IBDDE_4$
Ackley	∞	$3.17e+03$	$3.00e+03$	$\mathbf{2.96e+03}$	$4.06e+03$
Alpine	∞	$4.29e+03$	$\mathbf{3.28e+03}$	$3.29e+03$	∞
Ax.-par. hyp.-ell.	$2.81e+03$	$1.88e+03$	$\mathbf{1.72e+03}$	$1.72e+03$	$2.34e+03$
DeJong	$2.84e+03$	$1.78e+03$	$\mathbf{1.74e+03}$	$1.77e+03$	$2.37e+03$
DropWave	$2.45e+05$	$7.52e+03$	$\mathbf{6.02e+03}$	$1.39e+04$	$1.23e+05$
Griewangk	$2.84e+03$	$1.88e+03$	$\mathbf{1.74e+03}$	$1.75e+03$	$2.37e+03$
Michalewicz	∞	$3.56e+04$	∞	$\mathbf{8.97e+03}$	∞
Pathological	∞	$\mathbf{9.52e+03}$	∞	$2.60e+04$	∞
Rastrigin	∞	$1.21e+04$	$1.11e+04$	$\mathbf{3.73e+03}$	∞
Rosenbrock	$2.43e+03$	$1.37e+03$	$\mathbf{1.28e+03}$	$1.40e+03$	$1.94e+03$
Schwefel	∞	$1.69e+04$	∞	$\mathbf{3.70e+03}$	∞
Sum of powers	$3.66e+02$	$3.25e+02$	$\mathbf{3.17e+02}$	$3.40e+02$	$3.44e+02$
Tirronen	$5.80e+03$	$5.19e+03$	$5.30e+03$	$5.49e+03$	$\mathbf{5.05e+03}$
Rt. Ackley	∞	$3.92e+03$	$3.70e+03$	$\mathbf{3.53e+03}$	$8.24e+04$
Rt. Alpine	∞	$5.66e+03$	$\mathbf{3.97e+03}$	$7.65e+03$	∞
Rt. Ax.-par. hyp.-ell.	$3.78e+03$	$2.05e+03$	$\mathbf{1.89e+03}$	$2.03e+03$	$3.03e+03$
Rt. Griewangk	$3.40e+03$	$2.10e+03$	$\mathbf{1.82e+03}$	$1.93e+03$	$2.70e+03$
Rt. Michalewicz	$6.56e+03$	$4.55e+03$	$\mathbf{4.32e+03}$	$5.02e+03$	$5.54e+03$
Rt. Pathological	$4.48e+03$	$\mathbf{4.33e+03}$	$4.36e+03$	$4.34e+03$	$4.39e+03$
Rt. Rastrigin	∞	$6.67e+03$	$\mathbf{5.12e+03}$	$1.43e+04$	∞
Rt. Rosenbrock	$2.91e+03$	$1.66e+03$	$\mathbf{1.49e+03}$	$1.65e+03$	$2.43e+03$
Rt. Schwefel	∞	$6.76e+03$	$\mathbf{4.58e+03}$	$1.29e+04$	∞
Rt. Sum of powers	$4.16e+00$	$4.20e+00$	$\mathbf{4.04e+00}$	$4.12e+00$	$4.24e+00$
Rt. Tirronen	$5.14e+03$	$5.23e+03$	$\mathbf{4.76e+03}$	$4.98e+03$	$5.48e+03$

Analogous to PDE and IBDDE, the DDE integrating the schemes 1, 2, 3, and 4 is indicated with DDE_1, DDE_2, DDE_3, and DDE_4. The parameter g_r in DDE_3 has been set equal to 30. Table 8 lists final average fitness values over 50 independent runs and respective standard deviation values. Table 9 and 10 show the results of the Wilcoxon rank-sum test and Q-test respectively. The best results are highlighted in bold face.

Some performance trends (averaged over 50 independent runs) of the DDE (solid line) and its multiple scale factors variants (dashed and dotted lines) are displayed in Fig. 10.

Numerical results show that employment of multiple scale factors is not beneficial for the DDE structure. It can be noticed from Tables 8 and 9 that the standard DDE is ranked first in seventeen out of the twenty-four cases and in ten of these, its performance cannot be statistically distinguished from on or more of the variant algorithms. The use of multiple scale factor values therefore seems detrimental to the performance of the DDE algorithm. The Q-test results listed in Table 10 confirm that observation: the original DDE algorithm obtains the best result in sixteen cases out of twenty-four. One must also notice the fact that the table presents no ∞ symbol, indicating that the five algorithms are all robust, but also that their performances are very similar, which confirms the performance analysis based on Table 9.

The fitness trends presented in Fig. 10 are typical of the DDE algorithm: the algorithm improves very much upon its initial solutions for a brief period of time, and is followed by a a much longer period where only marginal improvement is obtained. Figure 10c illustrates a case where Scheme 4 presents the best performance, closely seconded by the other schemes, while the original DDE exhibit poorer peformance. On the contrary, Fig. 10a and b are representative of the case where the original DDE outperforms its variants. In the case of Fig. 10a,

Table 8 Average final fitness values ± standard deviations for DDE and its multiple scale factors variants

	DDE	DDE_1	DDE_2	DDE_3	DDE_4
Ackley	$\mathbf{2.13e+00 \pm 1.46e-01}$	$2.23e+00 \pm 1.64e-01$	$2.21e+00 \pm 1.32e-01$	$2.19e+00 \pm 1.81e-01$	$2.25e+00 \pm 1.35e-01$
Alpine	$5.00e+02 \pm 4.00e+01$	$5.09e+02 \pm 5.17e+01$	$\mathbf{5.00e+02 \pm 4.37e+01}$	$5.27e+02 \pm 4.75e+01$	$5.07e+02 \pm 4.47e+01$
Ax.-par. hyp.-ell.	$\mathbf{1.97e+05 \pm 2.82e+04}$	$2.28e+05 \pm 3.77e+04$	$2.28e+05 \pm 3.17e+04$	$2.29e+05 \pm 3.12e+04$	$2.29e+05 \pm 3.41e+04$
DeJong	$\mathbf{9.07e+02 \pm 1.11e+02}$	$9.51e+02 \pm 1.36e+02$	$1.00e+03 \pm 1.42e+02$	$1.00e+03 \pm 1.17e+02$	$9.93e+02 \pm 1.08e+02$
DropWave	$\mathbf{-1.31e-03 \pm 7.69e-05}$	$-1.26e-03 \pm 6.91e-05$	$-1.27e-03 \pm 5.40e-05$	$-1.26e-03 \pm 8.79e-05$	$-1.25e-03 \pm 5.38e-05$
Griewangk	$\mathbf{3.59e+03 \pm 4.20e+02}$	$3.87e+03 \pm 4.98e+02$	$3.97e+03 \pm 5.01e+02$	$3.86e+03 \pm 5.01e+02$	$3.87e+03 \pm 4.05e+02$
Michalewicz	$-2.07e+02 \pm 1.25e+01$	$\mathbf{-2.10e+02 \pm 1.06e+01}$	$-2.09e+02 \pm 1.08e+01$	$-2.08e+02 \pm 1.23e+01$	$-2.09e+02 \pm 1.32e+01$
Pathological	$-8.50e+01 \pm 8.18e+00$	$-8.84e+01 \pm 9.39e+00$	$-8.74e+01 \pm 8.76e+00$	$-8.64e+01 \pm 9.85e+00$	$\mathbf{-9.08e+01 \pm 9.38e+00}$
Rastrigin	$\mathbf{4.09e+03 \pm 2.80e+02}$	$4.19e+03 \pm 2.89e+02$	$4.13e+03 \pm 2.65e+02$	$4.15e+03 \pm 2.56e+02$	$4.27e+03 \pm 3.09e+02$
Rosenbrock	$\mathbf{2.77e+04 \pm 4.64e+03}$	$3.02e+04 \pm 4.87e+03$	$2.96e+04 \pm 5.12e+03$	$2.80e+04 \pm 5.37e+03$	$2.99e+04 \pm 5.38e+03$
Schwefel	$-8.20e+04 \pm 5.12e+03$	$\mathbf{-8.86e+04 \pm 5.88e+03}$	$-8.72e+04 \pm 4.39e+03$	$-8.77e+04 \pm 6.40e+03$	$-8.76e+04 \pm 5.71e+03$
Sum of powers	$2.96e-03 \pm 9.71e-03$	$\mathbf{1.54e-03 \pm 3.03e-03}$	$1.67e-03 \pm 3.14e-03$	$3.06e-03 \pm 9.66e-03$	$2.36e-03 \pm 6.40e-03$
Tirronen	$-2.90e-01 \pm 6.40e-02$	$-3.15e-01 \pm 7.41e-02$	$\mathbf{-3.22e-01 \pm 7.03e-02}$	$-3.08e-01 \pm 7.46e-02$	$-3.04e-01 \pm 7.81e-02$
Rt. Ackley	$\mathbf{2.20e+00 \pm 1.36e-01}$	$2.35e+00 \pm 1.62e-01$	$2.31e+00 \pm 1.07e-01$	$2.27e+00 \pm 1.48e-01$	$2.34e+00 \pm 1.34e-01$
Rt. Alpine	$5.59e+02 \pm 4.19e+01$	$5.74e+02 \pm 4.42e+01$	$5.75e+02 \pm 4.14e+01$	$5.80e+02 \pm 4.18e+01$	$\mathbf{5.56e+02 \pm 4.53e+01}$
Rt. Ax.-par. hyp.-ell.	$\mathbf{1.88e+05 \pm 2.78e+04}$	$2.12e+05 \pm 3.13e+04$	$2.18e+05 \pm 2.95e+04$	$2.13e+05 \pm 2.62e+04$	$2.11e+05 \pm 2.65e+04$
Rt. Griewangk	$\mathbf{3.28e+03 \pm 3.74e+02}$	$3.67e+03 \pm 3.89e+02$	$3.68e+03 \pm 4.27e+02$	$3.70e+03 \pm 4.83e+02$	$3.58e+03 \pm 4.33e+02$
Rt. Michalewicz	$-1.04e+02 \pm 5.63e+00$	$-1.09e+02 \pm 5.77e+00$	$-1.09e+02 \pm 6.42e+00$	$-1.09e+02 \pm 5.44e+00$	$\mathbf{-1.11e+02 \pm 6.26e+00}$
Rt. Pathological	$-6.17e+01 \pm 5.30e+00$	$-6.58e+01 \pm 5.55e+00$	$-6.65e+01 \pm 5.38e+00$	$-6.56e+01 \pm 4.51e+00$	$\mathbf{-6.65e+01 \pm 4.19e+00}$
Rt. Rastrigin	$\mathbf{4.54e+03 \pm 2.26e+02}$	$4.57e+03 \pm 2.42e+02$	$4.56e+03 \pm 2.16e+02$	$4.55e+03 \pm 2.35e+02$	$4.55e+03 \pm 2.26e+02$
Rt. Rosenbrock	$\mathbf{2.60e+04 \pm 5.35e+03}$	$3.00e+04 \pm 4.96e+03$	$2.90e+04 \pm 5.28e+03$	$2.86e+04 \pm 5.09e+03$	$2.98e+04 \pm 5.45e+03$
Rt. Schwefel	$-7.52e+04 \pm 4.18e+03$	$\mathbf{-7.90e+04 \pm 3.46e+03}$	$-7.88e+04 \pm 3.83e+03$	$-7.80e+04 \pm 4.27e+03$	$-7.78e+04 \pm 4.30e+03$
Rt. Sum of powers	$4.70e-02 \pm 1.36e-01$	$4.60e-02 \pm 1.03e-01$	$5.11e-02 \pm 1.20e-01$	$\mathbf{3.14e-02 \pm 1.24e-01}$	$3.77e-02 \pm 9.21e-02$
Rt. Tirronen	$-3.46e-01 \pm 6.94e-02$	$-3.56e-01 \pm 5.86e-02$	$\mathbf{-3.82e-01 \pm 8.14e-02}$	$-3.57e-01 \pm 6.31e-02$	$-3.75e-01 \pm 6.50e-02$

Table 9 Results of the Wilcoxon rank-sum test for the DDE and its variants

	1	2	3	4	5
Ackley	DDE DDE_3		DDE_1 DDE_2 DDE_4		
Alpine	DDE DDE_1 DDE_2 DDE_4				DDE_3
Ax.-par. hyp.-ell.	DDE	DDE_1 DDE_2 DDE_3 DDE_4			
DeJong	DDE DDE_1		DDE_2 DDE_3 DDE_4		
DropWave	DDE	DDE_1 DDE_2 DDE_3 DDE_4			
Griewangk	DDE	DDE_1 DDE_2 DDE_3 DDE_4			
Michalewicz	DDE DDE_1 DDE_2 DDE_3 DDE_4				
Pathological	DDE_4	DDE DDE_1 DDE_2 DDE_3			
Rastrigin	DDE DDE_1 DDE_2 DDE_3				DDE_4
Rosenbrock	DDE DDE_2 DDE_3			DDE_1 DDE_4	
Schwefel	DDE_1 DDE_2 DDE_3 DDE_4				DDE
Sum of powers	DDE DDE_1 DDE_2 DDE_3 DDE_4				
Tirronen	DDE_2	DDE DDE_1 DDE_3 DDE_4			
Rt. Ackley	DDE	DDE_3	DDE_1 DDE_2 DDE_4		
Rt. Alpine	DDE DDE_1 DDE_2 DDE_4				DDE_3
Rt. Ax.-par. hyp.-ell.	DDE	DDE_1 DDE_2 DDE_3 DDE_4			
Rt. Griewangk	DDE	DDE_1 DDE_2 DDE_3 DDE_4			
Rt. Michalewicz	DDE_1 DDE_2 DDE_3 DDE_4				DDE
Rt. Pathological	DDE_1 DDE_2 DDE_3 DDE_4				DDE
Rt. Rastrigin	DDE DDE_1 DDE_2 DDE_3 DDE_4				
Rt. Rosenbrock	DDE	DDE_1 DDE_2 DDE_3 DDE_4			
Rt. Schwefel	DDE_1 DDE_2 DDE_3 DDE_4				DDE
Rt. Sum of powers	DDE DDE_1 DDE_2 DDE_3 DDE_4				
Rt. Tirronen	DDE_2 DDE_4		DDE DDE_1 DDE_3		

Table 10 Results of the Q-test for the DDE and its variants

	DDE	DDE_1	DDE_2	DDE_3	DDE_4
Ackley	**1.37*e*+03**	2.59*e*+03	2.42*e*+03	1.98*e*+03	3.78*e*+03
Alpine	**8.83*e*+02**	1.23*e*+03	9.35*e*+02	1.29*e*+03	1.01*e*+03
Ax.-par. hyp.-ell.	**8.04*e*+02**	1.38*e*+03	2.10*e*+03	1.77*e*+03	1.61*e*+03
DeJong	**7.52*e*+02**	1.37*e*+03	1.58*e*+03	1.60*e*+03	1.50*e*+03
DropWave	**3.74*e*+02**	8.22*e*+02	7.26*e*+02	8.23*e*+02	1.20*e*+03
Griewangk	**7.30*e*+02**	1.34*e*+03	1.76*e*+03	1.46*e*+03	1.49*e*+03
Michalewicz	1.61*e*+03	**1.20*e*+03**	1.30*e*+03	1.50*e*+03	1.33*e*+03
Pathological	2.30*e*+03	1.67*e*+03	1.92*e*+03	1.77*e*+03	**1.32*e*+03**
Rastrigin	**9.78*e*+02**	1.34*e*+03	1.14*e*+03	1.24*e*+03	1.69*e*+03
Rosenbrock	**5.70*e*+02**	7.43*e*+02	8.55*e*+02	6.31*e*+02	7.90*e*+02
Schwefel	3.08*e*+03	**1.07*e*+03**	1.23*e*+03	1.40*e*+03	1.14*e*+03
Sum of powers	**9.89*e*+01**	1.03*e*+02	1.03*e*+02	1.08*e*+02	1.00*e*+02
Tirronen	7.40*e*+02	5.41*e*+02	**4.84*e*+02**	5.24*e*+02	6.00*e*+02
Rt. Ackley	**1.34*e*+03**	3.23*e*+03	3.57*e*+03	2.15*e*+03	3.13*e*+03
Rt. Alpine	**9.55*e*+02**	1.04*e*+03	1.13*e*+03	1.40*e*+03	1.16*e*+03
Rt. Ax.-par. hyp.-ell.	**7.38*e*+02**	1.60*e*+03	1.54*e*+03	1.44*e*+03	1.41*e*+03
Rt. Griewangk	**7.73*e*+02**	1.57*e*+03	1.67*e*+03	1.85*e*+03	1.37*e*+03
Rt. Michalewicz	1.96*e*+03	1.10*e*+03	1.61*e*+03	**1.02*e*+03**	1.05*e*+03
Rt. Pathological	1.95*e*+03	1.10*e*+03	1.07*e*+03	1.17*e*+03	**9.35*e*+02**
Rt. Rastrigin	**8.45*e*+02**	9.70*e*+02	1.08*e*+03	1.03*e*+03	1.11*e*+03
Rt. Rosenbrock	**5.70*e*+02**	9.61*e*+02	7.09*e*+02	7.02*e*+02	1.02*e*+03
Rt. Schwefel	2.03*e*+03	**1.03*e*+03**	1.05*e*+03	1.03*e*+03	1.48*e*+03
Rt. Sum of powers	**4.00*e*+00**	4.00*e*+00	4.20*e*+00	4.16*e*+00	4.00*e*+00
Rt. Tirronen	7.84*e*+02	7.60*e*+02	**4.56*e*+02**	9.18*e*+02	5.90*e*+02

the performance of DDE is not significantly better than the one of the other variants, while Fig. 10b is an example of the opposite being true.

A comparison with the performance of the DDE algorithm employing the binomial crossover presented in Weber et al. (2011) shows that multiple scale factors are not beneficial to DDE when combined with the exponential crossover. This result is the opposite of the previous study (with binomial crossover), which had shown that Scheme 4 offered a clear improvement on DDE, which is not the case anymore in the present study. In the current case, the higly exploitative character of the exponential crossover is accentuated by the small sizes of the subpopulations, and the limited amounts of available exploratory moves. The numerical results in this paper show that the combined effects of those two characteristics cannot be balanced by the use of multiple scale factors.

7 Conclusion

This paper studies the effect of four multiple scale factors schemes on distributed DE structures employing the exponential crossover. These four schemes have been applied to three

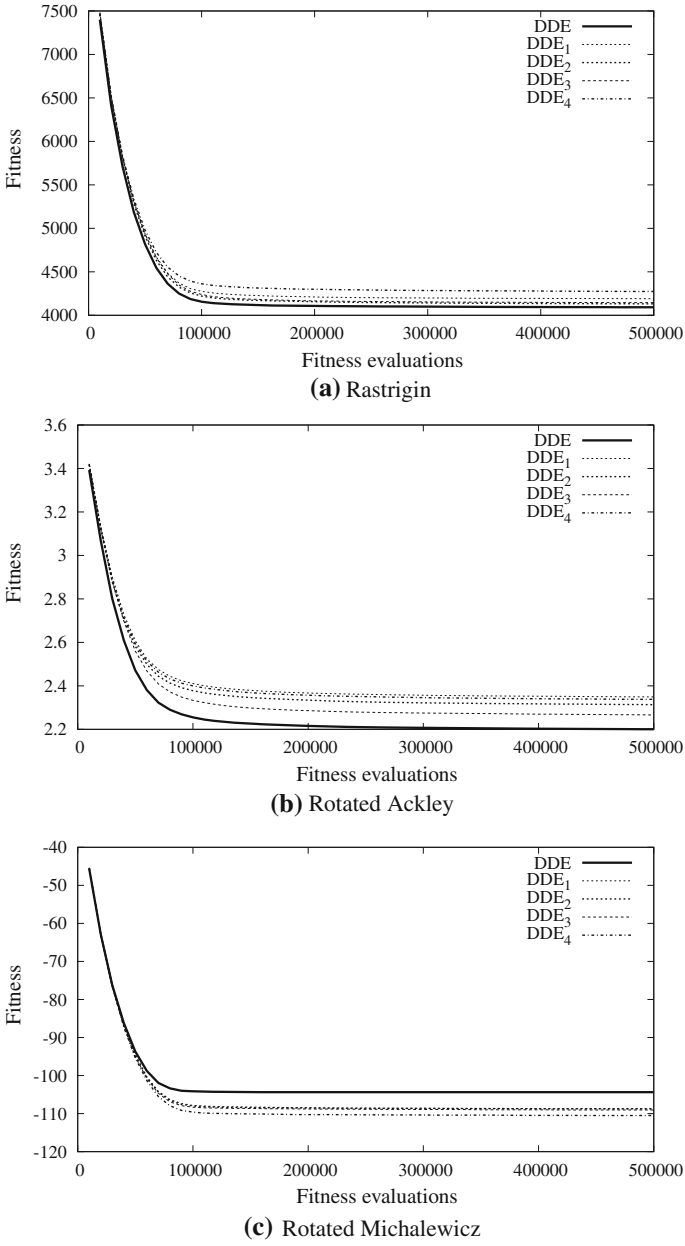


Fig. 10 Performance trends of DDE and its variants

modern distributed algorithms, namely PDE, IBDDE, and DDE. All these schemes use different values of the scale factor in the different subpopulations of the distributed DE, but while the two first schemes (1 and 2) use static values which remain constant during the evolution, the two other (3 and 4) update the values of the scale factor in the course of the optimization process.

Numerical results show that while schemes 2 and 3 are beneficial to the IBDDE structure, multiple values of the scale factor have a neutral effect on PDE and a detrimental effect on DDE. These results go agains those obtained in a previous study, where the binomial crossover was employed, and which had shown that all schemes were beneficial to the distributed structures, and schemes 2, 3 and 4 were especially beneficial for PDE, IBDDE and DDE, respectively.

In other words, multiple scale factors lead to an important improvement when associated to the binomial crossover, but is not necessarily improving distributed DE structures employing the exponential crossover, due to the much more exploitative characteristic of DE when using the exponential crossover which cannot necessarily be balanced by the use if multiple scale factors. As a more general corollary, algorithmic enhancements that improve binomial-crossover based algorithms do not necessarily improve exponential-crossover based algorithms: it is therefore a wrong approach to expect improvements by blindly combining two algorithms that, separately, produce good results.

Acknowledgments This work is supported by Academy of Finland, Akatemiatutkija 130600, Algorithmic Design Issues in Memetic Computing

References

Alba E, Tomassini M (2002) Parallelism and evolutionary algorithms. IEEE Trans Evol Comput 6(5):443–462

Apolloni J, Leguizamón G, García-Nieto J, Alba E (2008) Island based distributed differential evolution: an experimental study on hybrid testbeds. In: Proceedings of the IEEE international conference on hybrid intelligent systems, pp 696–701

Brest J, Maučec MS (2008) Population size reduction for the differential evolution algorithm. Appl Intell 29(3):228–247

Brest J, Greiner S, Bošković B, Mernik M, Žumer 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, Zamuda A, Bošković B, Maucec MS, Žumer V (2008) High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction. In: Proceedings of the IEEE world congress on computational intelligence, pp 2032–2039

Caponio A, Neri F (2009) Differential evolution with noise analysis. In: Applications of evolutionary Computing, lecture notes in computer science, vol 5484. Springer, pp 715–724

Chakraborty UK (ed) (2008) Advances in differential evolution, studies in computational intelligence, vol 143. Springer, Berlin

Das S, Konar A, Chakraborty UK (2005) Two improved differential evolution schemes for faster global search. In: Proceedings of the 2005 conference on genetic and evolutionary computation. ACM, New York, pp 991–998

De Falco I, Della Cioppa A, Maisto D, Scafuri U, Tarantino E (2007a) Satellite image registration by distributed differential evolution. In: Applications of evolutionary computing, lectures notes in computer science, vol 4448, Springer, pp 251–260

De Falco I, Maisto D, Scafuri U, Tarantino E, Della Cioppa A (2007b) Distributed differential evolution for the registration of remotely sensed images. In: Proceedings of the IEEE euromicro international conference on parallel, distributed and network-based processing, pp 358–362

De Falco I, Scafuri U, Tarantino E, Della Cioppa A (2007c) A distributed differential evolution approach for mapping in a grid environment. In: Proceedings of the IEEE euromicro international conference on parallel, distributed and network-based processing, pp 442–449

Fan HY, Lampinen J (2003) A trigonometric mutation operation to differential evolution. J Glob Optim 27(1):105–129

Feoktistov V (2006) Differential evolution in search of solutions. Springer, Berlin

Gämperle R, Müller SD, Koumoutsakos P (2002) A parameter study for differential evolution. In: Proceedings of the conference in neural networks and applications (NNA), fuzzy sets and fuzzy systems (FSFS) and evolutionary computation (EC), WSEAS, pp 293–298

Herrera F, Lozano M, Molina D (2010) Components and parameters of de, real-coded chc, and g-cmaes. Web document. http://sci2s.ugr.es/eamhco/descriptions.pdf
Karaboga D, Akay B (2009) A survey: algorithms simulating bee swarm intelligence. Artif Intell Rev 31 (1–4):61–85
Kozlov KN, Samsonov AM (2006) New migration scheme for parallel differential evolution. In: Proceedings of the international conference on bioinformatics of genome regulation and structure, pp 141–144
Kwedlo W, Bandurski K (2006) A parallel differential evolution algorithm. In: Proceedings of the IEEE international symposium on parallel computing in electrical engineering, pp 319–324
Lampinen J (1999) Differential evolution—new naturally parallel approach for engineering design optimization. In: Topping BH (ed) Developments in computational mechanics with high performance computing. Civil-Comp Press, UK pp 217–228
Lampinen J, Zelinka I (2000) On stagnation of the differential evolution algorithm. In: Oŝmera P (ed) Proceedings of 6th international mendel conference on soft computing, pp 76–83
Liu J, Lampinen J (2002a) A fuzzy adaptive differential evolution algorithm. In: Proceedings of the 17th IEEE region 10 international conference on computer, communications, control and power engineering, vol 1, pp 606–611
Liu J, Lampinen J (2002b) On setting the control parameter of the differential evolution algorithm. In: Proceedings of the 8th international mendel conference on soft computing, pp 11–18
Mallipeddi R, Suganthan PN (2008) Empirical study on the effect of population size on differential evolution algorithm. In: Proceedings of the IEEE congress on evolutionary computation, pp 3663–3670
Neri F, Tirronen V (2008) On memetic differential evolution frameworks: a study of advantages and limitations in hybridization. In: Proceedings of the IEEE world congress on computational intelligence, pp 2135–2142
Neri F, Tirronen V (2009) Scale factor local search in differential evolution. Memet Comput 1(2):153–171
Neri F, Tirronen V (2010) Recent advances in differential evolution: a review and experimental analysis. Artif Intell Rev 33(1):61–106
Neri F, Tirronen V, Kärkkäinen T (2009) Enhancing differential evolution frameworks by scale factor local search—part II. In: Proceedings of the IEEE congress on evolutionary computation, pp 118–125
Nipteni MS, Valakos I, Nikolos I (2006) An asynchronous parallel differential evolution algorithm. In: Proceedings of the ERCOFTAC conference on design optimisation: methods and application
Noman N, Iba H (2005) Enhancing differential evolution performance with local search for high dimensional function optimization. In: Proceedings of the 2005 conference on genetic and evolutionary computation. ACM, New York, pp 967–974
Olorunda O, Engelbrecht A (2007) Differential evolution in high-dimensional search spaces. In: Proceedings of the IEEE congress on evolutionary computation, pp 1934–1941
Omran MG, Salman A, Engelbrecht AP (2005) Self-adaptive differential evolution. In: Computational intelligence and security, lecture notes in computer science, vol 3801. Springer, Berlin, pp 192–199
Pavlidis NG, Tasoulis DK, Plagianakos VP, Nikiforidis G, Vrahatis MN (2005) Spiking neural network training using evolutionary algorithms. In: Proceedings of the IEEE international joint conference on neural networks, pp 2190–2194
Price K, Storn R (1997) Differential evolution: a simple evolution strategy for fast optimization. Dr Dobb's J Softw Tools 22(4):18–24
Price KV (1999) Mechanical engineering design optimization by differential evolution. In: Corne D, Dorigo M, Glover F (eds) New ideas in optimization. McGraw-Hill, New Delhi pp 293–298
Price KV, Storn R, Lampinen J (2005) Differential evolution: a practical approach to global optimization. Springer, Berlin
Qin AK, Suganthan PN (2005) Self-adaptive differential evolution algorithm for numerical optimization. In: Proceedings of the IEEE congress on evolutionary computation, vol 2, pp 1785–1791
Qin AK, Huang VL, Suganthan PN (2009) Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans Evol Comput 13:398–417
Rönkkönen J, Lampinen J (2003) On using normally distributed mutation step length for the differential evolution algorithm. In: Matousek R, Osmera P (eds) Proceedings of ninth international MENDEL conference on soft computing, pp 11–18
Rönkkönen J, Kukkonen S, Price KV (2005) Real-parameter optimization with differential evolution. In: Proceedings of IEEE international conference on evolutionary computation, vol 1, pp 506–513
Salman A, Engelbrecht AP, Omran MG (2007) Empirical analysis of self-adaptive differential evolution. Eur J Oper Res 183(2):785–804
Salomon M, Perrin GR, Heitz F, Armspach JP (2005) Parallel differential evolution: Application to 3-d medical image registration. In: Price KV, Storn RM, Lampinen JA (eds) Differential evolution–a practical approach to global optimization chap 7, natural computing series. Springer, Berlin, pp 353–411

Soliman OS, Bui LT (2008) A self-adaptive strategy for controlling parameters in differential evolution. In: Proceedings of the IEEE congress on evolutionary computation, pp 2837–2842
Soliman OS, Bui LT, Abbass HA (2007) The effect of a stochastic step length on the performance of the differential evolution algorithm. In: Proceedings of the IEEE congress on evolutionary computation, pp 2850–2857
Storn R (1999) System design by constraint adaptation and differential evolution.. IEEE Trans Evol Comput 3(1):22–34
Storn R, Price K (1995) Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, ICSI
Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11:341–359
Tasoulis DK, Pavlidis NG, Plagianakos VP, Vrahatis MN (2004) Parallel differential evolution. In: Proceedings of the IEEE congress on evolutionary computation, pp 2023–2029
Tirronen V, Neri F, Rossi T (2009) Enhancing differential evolution frameworks by scale factor local search—part I. In: Proceedings of the IEEE congress on evolutionary computation, pp 94–101
Weber M, Neri F, Tirronen V (2009) Distributed differential evolution with explorative-exploitative population families. Genet Program Evolv Mach 10(4):343–371
Weber M, Neri F, Tirronen V (2010) Shuffle or update parallel differential evolution for large-scale optimization. Soft Comput (To appear)
Weber M, Neri F, Tirronen V (2011) A study on scale factor in distributed differential evolution. Inf Sci 181:2488–2511
Wilcoxon F (1945) Individual comparisons by ranking methods. Biom Bull 1(6):80–83
Zaharie D (2002a) Critical values for control parameters of differential evolution algorithm. In: Matuŝek R, Oŝmera P (eds) Proceedings of 8th international mendel conference on soft computing, pp 62–67
Zaharie D (2002b) Parameter adaptation in differential evolution by controlling the population diversity. In: D Petcu et al. (eds) Proceedings of the international workshop on symbolic and numeric algorithms for scientific computing, pp 385–397
Zaharie D (2003) Control of population diversity and adaptation in differential evolution algorithms. In: Matoušek R, Ošmera P (eds) Proceedings of MENDEL international conference on soft computing, pp 41–46
Zaharie D, Petcu D (2003) Parallel implementation of multi-population differential evolution. In: Proceedings of the NATO advanced research workshop on concurrent information processing and computing. IOS Press, Amsterdam, pp 223–232
Zhang J, Sanderson AC (2009) Jade: Adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958
Zielinski K, Weitkemper P, Laur R, Kammeyer KD (2006) Parameter study for differential evolution using a power allocation problem including interference cancellation. In: Proceedings of the IEEE congress on evolutionary computation, pp 1857–1864