Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 287607, 10 pages
http://dx.doi.org/10.1155/2015/287607

Research Article

Differential Evolution with Population and Strategy Parameter Adaptation

V. Gonuguntla, R. Mallipeddi, and Kalyana C. Veluvolu

School of Electronics Engineering, Kyungpook National University, Daegu 702 701, Republic of Korea

Correspondence should be addressed to R. Mallipeddi; mallipeddi.ram@gmail.com

Received 24 November 2014; Accepted 26 February 2015

Academic Editor: Francesco Ubertini

Differential evolution (DE) is simple and effective in solving numerous real-world global optimization problems. However, its effectiveness critically depends on the appropriate setting of population size and strategy parameters. Therefore, to obtain optimal performance the time-consuming preliminary tuning of parameters is needed. Recently, different strategy parameter adaptation techniques, which can automatically update the parameters to appropriate values to suit the characteristics of optimization problems, have been proposed. However, most of the works do not control the adaptation of the population size. In addition, they try to adapt each strategy parameters individually but do not take into account the interaction between the parameters that are being adapted. In this paper, we introduce a DE algorithm where both strategy parameters are self-adapted taking into account the parameter dependencies by means of a multivariate probabilistic technique based on Gaussian Adaptation working on the parameter space. In addition, the proposed DE algorithm starts by sampling a huge number of sample solutions in the search space and in each generation a constant number of individuals from huge sample set are adaptively selected to form the population that evolves. The proposed algorithm is evaluated on 14 benchmark problems of CEC 2005 with different dimensionality.

1. Introduction

Like most of the stochastic numerical optimization algorithms, differential evolution (DE) [1] starts with randomly sampled solution vectors which evolve over the generations with the help of genetic operators such as mutation, crossover, and selection. Due to its effectiveness, DE has been successfully employed to solve numerous optimization problems in various fields of engineering as communication [1], optics [2], and power systems [3].

However, experimentally [4] and theoretically [5] it has been demonstrated that the performance of DE is sensitive to the selection of mutation, crossover strategies, and their associated parameters such as, crossover rate (CR) and scale factor (F) and the population size (NP). In other words, the optimal combination of population size, strategies, and their associated control parameters can be different for different optimization problems. In addition, for the same optimization problem the optimal combination can vary depending on the available computational resources and accuracy requirements [6]. Therefore, to successfully solve a specific optimization problem, it is necessary to perform trial-and-error search for the most appropriate combination of population size, strategies, and their associated parameter values. However, the trial-and-error search process is time consuming and incurs high computational costs. Therefore, to overcome the time-consuming trial-and-error procedure different adaptation schemes such as SaDE [7] and JADE [8] have been proposed in the literature.

From the literature on adaptive/self-adaptive parameter control techniques, it is clear that it can be observed that even a moderate parameter adaptation scheme is much better than a well-tuned combination of individual parameters on a given set of benchmark problems. In addition, adaptive/self-adaptive technique can enhance the robustness by dynamically adapting the parameters to the characteristic of different fitness landscapes. In other words, a well-designed parameter adaptation technique can effectively solve various optimization problems without the need for the trial-and-error process of parameter tuning. In addition, the convergence rate can be improved if the control parameters are adapted to appropriate values at different evolution stages of a specific problem.

Unlike most parameter adaptation techniques in DE [7] that employ explorative mutation strategies to obtain better performance, the authors in [8] proposed a parameter adaptation method with a greedy mutation strategy and binomial crossover strategy as search basis. The proposed greedy mutation strategy "DE/current-to-*p*best" utilizes the information present in multiple best solutions to balance the greediness of the mutation and diversity of the population. In addition, the parameter adaptation technique is based on evolving the mutation factors and crossover probabilities based on their historical record of success.

In DE literature, most of the parameter adaptation techniques proposed consider the adaptation of the two different parameters, crossover probability and the scale factor, individually but do not consider the interaction between the two parameters. In other words, they do not take into account the side effects introduced by changing the values of the parameters individually. In addition, unlike the adaptation of strategies and their associated parameters, the adaptation of the population size in enhancing the performance of the DE algorithm has not been given significant consideration [9].

In this paper, we propose a parameter adaptation technique based on Gaussian Adaptation (GaA) [10], an estimation of distribution algorithm (EDA), to manage the dependencies between the two parameters (mutation scale factor, F, and the crossover probability, CR) considered. In addition, we propose a population adaptation scheme where the algorithm has a large set of sampled solutions which evolve over the generations. In each generation, a fixed number of individuals from the large set become the population members of the DE algorithm. The fixed number of individuals from the large set can be selected randomly or based on the objective value depending on the stage of evolution.

The reminder of this paper is organized as follows. Section 2 presents a literature survey on (1) DE and different adaptive DE variants and (2) Gaussian Adaptation. Section 3 presents the proposed algorithm. Section 4 presents the experimental results and discussions while Section 5 concludes the paper.

2. Literature Review

2.1. Differential Evolution. Differential Evolution (DE) [11], a parallel real-coded global optimization algorithm over continuous spaces, utilizes NP, D-dimensional parameter vectors, $\mathbf{X}_{i,G} = \{x_{i,G}^1, \ldots, x_{i,G}^D\}$, $i = 1, \ldots, NP$ uniformly sampled within the search space to encode the candidate solutions.

During every generation G, corresponding to each individual $\mathbf{X}_{i,G}$ in the current population, referred to as target vector, a mutant vector $\mathbf{V}_{i,G}$ is produced by one of the following mutation strategies:

"DE/best/1" [12]:

$$\mathbf{V}_{i,G} = \mathbf{X}_{\text{best},G} + F \cdot \left(\mathbf{X}_{r_1^i,G} - \mathbf{X}_{r_2^i,G}\right). \tag{1}$$

"DE/best/2" [12]:

$$\begin{aligned}\mathbf{V}_{i,G} = \mathbf{X}_{\text{best},G} &+ F \cdot \left(\mathbf{X}_{r_1^i,G} - \mathbf{X}_{r_2^i,G}\right) \\ &+ F \cdot \left(\mathbf{X}_{r_3^i,G} - \mathbf{X}_{r_4^i,G}\right).\end{aligned} \tag{2}$$

"DE/rand/1" [12]:

$$\mathbf{V}_{i,G} = \mathbf{X}_{r_1^i,G} + F \cdot \left(\mathbf{X}_{r_2^i,G} - \mathbf{X}_{r_3^i,G}\right). \tag{3}$$

"DE/current-to-rand/1" [13]:

$$\begin{aligned}\mathbf{U}_{i,G} = \mathbf{X}_{i,G} &+ K \cdot \left(\mathbf{X}_{r_1^i,G} - \mathbf{X}_{i,G}\right) \\ &+ F \cdot \left(\mathbf{X}_{r_2^i,G} - \mathbf{X}_{r_3^i,G}\right).\end{aligned} \tag{4}$$

The indices $r_1^i, r_2^i, r_3^i, r_4^i, r_5^i$ are mutually exclusive integers different from the index i and are randomly generated within the range $[1, NP]$. The scale factor F is a positive value, while K is randomly chosen within the range $[0, 1]$. $\mathbf{X}_{\text{best},G}$ is the solution vector with the best fitness value in the population at generation G.

After mutation, crossover operation is applied to each pair of the target vector $\mathbf{X}_{i,G}$ and its corresponding mutant vector $\mathbf{V}_{i,G}$ to generate a trial vector $\mathbf{U}_{i,G}$. In DE, the most commonly used crossover is the binomial (uniform) crossover defined as follows [11]:

$$u_{i,G}^j = \begin{cases} v_{i,G}^j & \text{if } \left(\text{rand}_j\,[0,1] \le \text{CR}\right) \text{ or } \left(j = j_{\text{rand}}\right) \\ x_{i,G}^j & \text{otherwise} \end{cases} \qquad j = 1, 2, \ldots, D. \tag{5}$$

The crossover rate CR is a user-specified constant within the range $[0, 1]$, while j_{rand} is a randomly chosen integer in the range $[1, D]$.

After the crossover, the trial vectors are evaluated to obtain the objective function and selection operation is performed. The objective function value of each trial vector $f(\mathbf{U}_{i,G})$ is compared to that of its corresponding target vector $f(\mathbf{X}_{i,G})$ in the current population. If the trial vector is better than the corresponding target vector, the trial vector will replace the target vector and enter the population of the next generation. In a minimization problem, the selection operation can be expressed as follows:

$$\mathbf{X}_{i,G+1} = \begin{cases} \mathbf{U}_{i,G}, & \text{if } f\left(\mathbf{U}_{i,G}\right) \le f\left(\mathbf{X}_{i,G}\right) \\ \mathbf{X}_{i,G}, & \text{otherwise.} \end{cases} \tag{6}$$

In DE, mutation, crossover, and selection are repeated generation after generation until a termination criterion is satisfied. The algorithmic description of DE is summarized as follows.

Differential Evolution Algorithm

Step 1. Set the generation number $G = 0$ and randomly initialize a population of NP individuals.

Step 2. WHILE stopping criterion is not satisfied.
DO.

Step 2.1. Mutation.

Step 2.2. Crossover.

Step 2.3. Selection.

Step 2.4. Increment the generation count $G = G + 1$.

Step 3. END WHILE.

As mentioned earlier, the performance of the conventional DE algorithm depends on the population size, chosen strategies, and their associated control parameters. In addition, as complexity of the optimization problem increases the performance of DE algorithm becomes more sensitive to the parameter settings [4]. Therefore, inappropriate choice of population size, mutation and crossover strategies, and their associated parameters may lead to premature convergence, stagnation, or wastage of computational resources [6]. In literature, various empirical guidelines were suggested for choosing the appropriate population size, strategies, and control parameter settings depending on the characteristics of the optimization problems [6]. However, depending on the complexity of the optimization problem, choosing an appropriate population size, strategies, and control parameters is not straight forward due to the complex interaction of control parameters with the DE's performance [14]. In addition, the manual setting and/or tuning of DE strategies and parameters based on the guidelines result in various conflicting conclusions, which lack sufficient justifications. Therefore, to avoid the tuning of parameters by trial-and-error procedure, various adaptive techniques have been proposed. Among the three parameters (*NP*, *F*, and CR), most of the parameter adaptive techniques except [9], set the population size (*NP*) to a predefined value based on the dimensionality of the problem.

Among the different adaptive DE variants, adaptive differential evolution proposed in [8], referred to as JADE, demonstrates good performance in terms of convergence speed and robustness on a variety of optimization problems. JADE [8] implements a mutation strategy "DE/current-to-*p*best" as a generalization to the classic "DE/current-to-best" strategy. Unlike the classic mutation strategy which uses the current best individual, "DE/current-to-*p*best" utilizes the information present in p fitter individuals of the current population. The use of multiple solutions helps in balancing the greediness of the mutation and the diversity of the population. In JADE, the control parameters (F and CR) are updated in an adaptive manner in order to alleviate the trial-and-error search. In JADE, using the "DE/current-to-*p*best", a mutation vector corresponding to the individual $\mathbf{X}_i$ in generation G is generated as

$$\begin{aligned}\mathbf{V}_{i,G} = \mathbf{X}_{i,G} + F_i\left(\mathbf{X}^{p}_{\text{best},G} - \mathbf{X}_{i,G}\right) \\ + F_i\left(\mathbf{X}_{r1,G} - \mathbf{X}_{r2,G}\right),\end{aligned} \tag{7}$$

where $\mathbf{X}_{r1,G}$, $\mathbf{X}_{r2,G}$, and $\mathbf{X}^{p}_{\text{best},G}$ are selected from the current population. At each generation, the scale factor F_i and crossover probability CR_i of each individual $\mathbf{X}_i$ are independently generated as

$$F_i = \text{rand}c_i\left(\mu_F, 0.1\right), \tag{8}$$

$$\text{CR}_i = \text{rand}n_i\left(\mu_{\text{CR}}, 0.1\right). \tag{9}$$

As shown in (8) and (9), the parameters F and CR corresponding to each individual are sampled using Cauchy and Normal distributions, respectively. Then mean values μ_F and μ_{CR} are initialized to 0.5 and are updated at the end of each generation as

$$\mu_F = (1 - c)\cdot\mu_F + c\cdot\text{mean}_L\left(S_F\right), \tag{10}$$

$$\mu_{\text{CR}} = (1 - c)\cdot\mu_{\text{CR}} + c\cdot\text{mean}_A\left(S_{\text{CR}}\right), \tag{11}$$

where c is a positive constant between 0 and 1. The terms $\text{mean}_A(\cdot)$ and $\text{mean}_L(\cdot)$ denote the arithmetic mean and Lehmer mean [8], respectively. S_F and S_{CR} denote the sets of mutation factors and crossover probabilities, respectively, that produced successful trial vectors in the previous generation.

During the past decade, hybridization of EAs has gained significance, due to ability to complement each other's strengths and overcome the drawbacks of the individual algorithms. In [15], the authors proposed a DE parameter adaptation technique based on harmony search (HS) algorithm in which a group of DE control parameter combinations are randomly initialized. The randomly initialized DE parameter combinations form the initial harmony memory (HM) of the HS algorithm. Each combination of the parameters present in the HM is evaluated by testing on the DE population during the evolution. Based on the effectiveness of the DE parameter combinations present in HM, the HS algorithm evolves the parameter combinations. At any given point of time during the evolution of the DE population, the HM contains an ensemble of DE parameters that suits the evolution process of the DE population.

As mentioned above, among the different adaptive DE variants [16–22], there exist only a few significant works considering the adaptation of the population size in DE algorithm [9]. In addition to the adaptation of control parameters F and CR, the author in [9] considered the self-adaptation of populations and the algorithm is referred to as DESAP. In DESAP the population size (*NP*) is automatically adapted from initialization to the completion of the evolution process. DESAP is proposed with two encoding methodologies: *absolute encoding methodology* and *relative encoding methodology*.

2.2. Gaussian Adaptation.

Optimization algorithms that rely on probabilistic models, such as Covariance Matrix Adaptation (CMA) [23] and Gaussian Adaptation (GaA) [10], belong to the class of Estimation of Distribution Algorithms (EDAs) and do not rely on any variation operators such as crossover or mutation. In EDAs, the most promising solutions of the previous generation are used to update the probability distribution model. The updated probability distribution model is used to sample the solutions for the next generation.

In other words, EDAs rely on the iterative random sampling and updating the probability distribution model in order to approximate the problem at hand. Therefore, the process in which the random samples are generated plays a crucial role. In continuous spaces, typical EDAs employ a multivariate Gaussian distribution as the probability density model [24]. Continuous optimization methods, such as GaA [10] and Evolution Strategies (ES) [23], use Gaussian sampling to generate candidate solutions from the target distribution and evaluates the target distribution at the sample points.

Covariance Matrix Adaptation (CMA-ES) [6] and GaA [10] constantly adapt the covariance matrix of the sampling distribution based on the previously accepted samples. In CMA-ES covariance adaptation is employed to increase the likelihood of generating successful mutations, while GaA adapts the covariance to maximize the entropy of the search distribution under the constraint that acceptable search points are found with a predefined, fixed hitting probability.

Gaussian Adaptation (GaA) is a stochastic process that adapts a Gaussian distribution to a region or set of feasible points in parameter space. As a result of the adaptation, GaA becomes a maximum dispersion process extending the sampling over the largest possible volume in parameter space while keeping the probability of finding feasible points at a suitable level. GaA is based on the principle of maximum entropy and tries to maximize the entropy H of a multivariate Gaussian distribution $N(\mathbf{m}, \mathbf{C})$ given the mean ($\mathbf{m}$) and the covariance ($\mathbf{C}$) information:

$$H = \log\left(\sqrt{(2\pi e)^D \det(\mathbf{C})}\right). \quad (12)$$

The entropy is maximized by maximizing the determinant of the covariance matrix.

The GaA algorithm starts with mean of a multivariate Gaussian distribution ($\mathbf{m}^{(0)}$) and an initial point ($\mathbf{x}^{(0)}$). In iteration ($g+1$), a new solution is sampled as

$$\mathbf{x}^{(g+1)} = \mathbf{m}^{(g)} + \mathbf{r}^{(g)}\mathbf{Q}^{(g)}\boldsymbol{\eta}^{(g)}, \quad (13)$$

where $\boldsymbol{\eta}^{(g)} \sim N(\mathbf{0}, \mathbf{I})$. $\mathbf{Q}^{(g)}$ is the normalized square root of $\mathbf{C}^{(g)}$ and is obtained by following decomposition:

$$\mathbf{C}^{(g)} = \left(\mathbf{r}\cdot\mathbf{Q}^{(g)}\right)\left(\mathbf{r}\cdot\mathbf{Q}^{(g)}\right)^T = \mathbf{r}^2\left(\mathbf{Q}^{(g)}\right)\left(\mathbf{Q}^{(g)}\right)^T, \quad (14)$$

where $\mathbf{r}$ is the scalar *step size.*

In order to minimize a real-valued objective function $f(\mathbf{x})$, GaA uses a fitness dependent acceptance threshold c_T which is monotonically lowered until some convergence criteria are met. If the objective value of the newly sampled solution in (13) is less than c_T, then the mean ($\mathbf{m}$), covariance ($\mathbf{C}$), and the scale factor ($\mathbf{r}$) are updated as follows:

$$\begin{gathered}\mathbf{m}^{(g+1)} = \left(1 - \frac{1}{N_m}\right)\mathbf{m}^{(g)} + \frac{1}{N_m}\mathbf{x}^{(g+1)}, \\ \mathbf{C}^{(g+1)} = \left(1 - \frac{1}{N_C}\right)\mathbf{C}^{(g)} + \frac{1}{N_C}(\Delta\mathbf{x})(\Delta\mathbf{x})^T, \\ \mathbf{r}^{(g+1)} = f_e \cdot \mathbf{r}^{(g)},\end{gathered} \quad (15)$$

where $f_e > 1$ is the expansion factor. N_m and N_C are the weighting factors. $\Delta\mathbf{x} = (\mathbf{x}^{(g+1)} - \mathbf{x}^{(g)})$.

If the objective value of the newly sampled solution $\mathbf{x}^{(g+1)}$ is greater than the threshold then the mean and covariance are not adapted but the step size is reduced as

$$\mathbf{r}^{(g+1)} = f_c \cdot \mathbf{r}^{(g)}, \quad (16)$$

where $f_c < 1$ is the contraction factor.

In order to use GaA for optimization, the acceptance threshold c_T is continuously lowered as follows:

$$c_T^{(g+1)} = \left(1 - \frac{1}{N_T}\right)c_T^{(g)} + \frac{1}{N_T}f\left(\mathbf{x}^{(g+1)}\right), \quad (17)$$

where N_T is the weighting factor. The fitness-dependent update of c_T makes the algorithm invariant to the linear transformations in the objective function.

Gaussian Adaptation Algorithm

Step 1. Set generation number $G = 0$. Initialize $\mathbf{m}$, $\mathbf{C}$, $\mathbf{r}$, and $c_{T.}$

Step 2. WHILE stopping criterion is not satisfied.
DO.

Step 2.1. Sample a new solution using (13).

Step 2.2. Evaluate and Check if the objective value of newly sampled solution is less the threshold c_T.

Step 2.3. Update $\mathbf{m}$, $\mathbf{C}$, $\mathbf{r}$, and c_T.

Step 2.4. Increment the generation count $G = G + 1$.

Step 3. END WHILE.

3. Proposed Differential Evolution Algorithm

As highlighted in the previous section, depending on the nature of problem (unimodal or multimodal) and available computation resources, different optimization problems require different population sizes, mutation, and crossover strategies combined with different parameter values to obtain optimal performance. In addition, to solve a specific problem, different mutation and crossover strategies with different parameter settings may be better during different stages of the evolution than a single set of strategies with unique parameter settings as in the conventional DE. Motivated by these observations, many adaptive and self-adaptive parameter adaptive techniques have been proposed [16–22]. As mentioned earlier most of the adaptive DE algorithms consider adapting scaling and crossover rate parameters only. In addition, most of the adaptive techniques which consider adapting the F and CR values adapt them individually. For instance, in JADE [8], the mutation factors and crossover probabilities are evolved based on their historical record of success. F and CR values corresponding to the individuals in the current generation are generated from corresponding mean values using Cauchy and Gaussian distributions, respectively. After the selection process, the F and CR values that were able to produce successful trial vectors are collected. Then the respective mean values of F and CR are updated using Lehmer and arithmetic means, respectively. In other words, the F and CR are generated (see (8) and (9)) and adapted (see (10) and (11)) individually. Therefore, JADE does not consider the intercorrelation between the two parameters. However, in [6], it has been demonstrated that performance of DE depends on the combination of F and CR. In other words, the parameters F and CR on which the performance of DE depends are intercorrelated. Therefore, adapting the two parameters individually may not result in the optimal performance of the DE algorithm.

In this paper, we present a population size, crossover rate, and scale factor adaptation. The proposed algorithm considers the intercorrelation between the two parameters, mutation scale factor, and crossover probability, during the process of adaptation. In other words, the parameters evolve based on the Gaussian adaptation process which is used for parameter optimization.

3.1. Parameter Adaptation Based on Gaussian Adaptation. Unlike most DE adaptation algorithms, the proposed algorithm adapts F and CR by employing GaA on the bidimensional continuous space composed of F and CR. Therefore, the data structures employed by the proposed algorithm are the mean vector $\mathbf{m}$ and the covariance matrix $\mathbf{C}$. The mean vector ($\mathbf{m}$) comprises the mean values of F and CR while the covariance matrix ($\mathbf{C}$) comprises the interdependencies between the two parameters.

As in JADE [8], every DE individual is assigned with a personal version of the parameters; that is, there is a couple F_i, CR_i for each individual i sampled using (13). In other words, every time that these parameters are needed (for mutation and crossover in DE), they are sampled from the multivariate Gaussian Distribution identified by $\mathbf{m}$ and $\mathbf{C}$. In the current work, the mean vector $\mathbf{m}$ is initialized to $[0.5, 0.5]$ and covariance matrix ($\mathbf{C}$) is set to an identity matrix.

During every generation of the DE evolution, the F_i and CR_i values corresponding to the individuals in the population are generated using the mean ($\mathbf{m}$) and the covariance matrix ($\mathbf{C}$) using (13). Each individual in the DE algorithm uses the F_i and CR_i values to produce the mutation vectors and consequently trial vectors. The combination of F_i and CR_i values that resulted in an offspring that produces maximum improvement is used to update the mean ($\mathbf{m}$) and the covariance ($\mathbf{C}$). The continuous updating of $\mathbf{m}$ and $\mathbf{C}$ by the parameter combinations that produced better solutions will help the parameter search to move to the regions where more suitable combination of the parameters can be generated. The limits of the F and CR are set to be $(0, 1.0]$ and $[0, 1.0]$, respectively.

3.2. Population Adaptation. In this work, the proposed algorithm starts by sampling a huge set of solutions within the search space. During every generation, the number of solutions equal to the population size (NP) is selected to evolve with the help of mutation, crossover, and selection operations. After the generation, during which the evolution of the individuals selected takes place, the members of the population are replaced into respective places in the huge set.

In the proposed algorithm, the selection of individuals from the huge set during every generation should be done in an appropriate manner to balance the exploration and exploitation. In this paper, we use a simple criterion to select the population individuals that can evolve based on objective value distribution of the individuals in the huge set.

The mean objective value corresponding to a set of individuals that are uniformly and randomly sampled from a search space will always be less than the median objective value of the set of individuals. As long as the mean objective value is less than the median objective value in the large set the population members for the next generation are selected using a tournament selection. The selection of individuals based on the tournament selection allows the fittest individuals to allow the population and evolve. However, when the mean objective value is greater than the median objective value it gives an indication that most of the solutions in the large set are within a basin. The solutions within a basin may lead to premature convergence and therefore to balance the exploration capability the population members for the next generation are taken at random from the large set.

Outline of Proposed DE Algorithm

Step 1. Set the generation count $G = 0$, and randomly sample $20 * NP$ solution vectors. Initialize $\mathbf{m}$, $\mathbf{C}$, $\mathbf{r}$, and c_T.

Step 2. WHILE stopping criterion is not satisfied.
DO.

Step 2.1. Select population members from the large set.

Step 2.2. Sample new parameter combinations using (14).

Step 2.3. Mutation.

Step 2.4. Crossover.

Step 2.5. Selection.

Step 2.6. Replace the solution vectors in the large set with the members of the current population at the respective indices.

Step 2.7. Check if the improvement by the best parameter combination is greater than the threshold c_T.

Step 2.8. Update $\mathbf{m}$, $\mathbf{C}$, $\mathbf{r}$, and c_T.

Step 2.9. Increment the generation count $G = G + 1$.

Step 3. END WHILE.

4. Experimental Setup and Results

We evaluated the performance of the proposed algorithm on a set of 14 test problems of CEC 2005 [25]. Each of the 14 test problems is scalable and the algorithm is tested on the 10D, 30D, 50D, and 100D versions of the test problems.

Table 1: Effect of population size on the performance of JADE algorithm (30D benchmark problems).

Fcn	*NP* = 50		*NP* = 100		*NP* = 200		*NP* = 400		*NP* = 600		*NP* = 800		*NP* = 1000	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
*F*1	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	4.78*E* − 19	3.97*E* − 19	3.59*E* − 13	1.87*E* − 13	1.49*E* − 09	4.44*E* − 10
*F*2	**1.24*E* − 27**	**6.95*E* − 28**	**1.79*E* − 28**	**1.50*E* − 28**	**2.00*E* − 27**	**7.16*E* − 27**	4.19*E* − 22	7.71*E* − 22	1.07*E* − 14	2.87*E* − 14	8.10*E* − 09	1.12*E* − 08	1.33*E* − 04	1.59*E* − 04
*F*3	7.99*E* + 03	4.76*E* + 03	1.70*E* + 04	1.19*E* + 04	4.35*E* + 03	5.05*E* + 03	**9.52*E* − 08**	**4.15*E* − 07**	2.86*E* − 05	6.61*E* − 05	2.03*E* − 01	3.75*E* − 01	1.59*E* + 02	1.78*E* + 02
*F*4	2.49*E* − 02	6.66*E* − 02	6.74*E* − 10	1.55*E* − 09	**8.93*E* − 17**	**1.60*E* − 16**	**5.64*E* − 18**	**1.68*E* − 17**	2.53*E* − 12	4.93*E* − 12	3.09*E* − 07	3.51*E* − 07	1.30*E* − 03	1.33*E* − 03
*F*5	9.15*E* + 02	4.50*E* + 02	2.47*E* + 02	1.68*E* + 02	6.30*E* + 01	5.48*E* + 01	**2.91*E* + 01**	**2.55*E* + 01**	4.63*E* + 01	4.17*E* + 01	2.89*E* + 02	1.27*E* + 02	6.64*E* + 02	1.55*E* + 02
*F*6	**4.06*E* + 00**	**1.47*E* + 01**	**4.46*E* + 00**	**1.88*E* + 01**	4.50*E* + 00	1.65*E* + 01	7.83*E* + 00	5.14*E* + 00	1.81*E* + 01	2.30*E* + 00	2.30*E* + 01	8.45*E* − 01	2.54*E* + 01	3.01*E* − 01
*F*7	2.04*E* − 02	1.44*E* − 02	1.41*E* − 02	1.33*E* − 02	7.06*E* − 03	9.33*E* − 03	2.22*E* − 03	4.68*E* − 03	1.40*E* − 03	4.65*E* − 03	1.00*E* − 03	3.04*E* − 03	**4.22*E* − 04**	**2.64*E* − 04**
*F*8	2.09*E* + 01	4.56*E* − 02	2.08*E* + 01	3.25*E* − 01	2.09*E* + 01	6.90*E* − 02	2.10*E* + 01	4.90*E* − 02	2.09*E* + 01	4.52*E* − 02	2.10*E* + 01	4.82*E* − 02	2.10*E* + 01	5.28*E* − 02
*F*9	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	**0.00*E* + 00**	1.00*E* − 11	4.74*E* − 12	1.14*E* − 01	2.02*E* − 02	8.56*E* + 00	8.33*E* − 01	2.04*E* + 01	1.43*E* + 00	3.11*E* + 01	2.05*E* + 00
*F*10	2.83*E* + 01	6.01*E* + 00	**2.58*E* + 01**	**5.13*E* + 00**	3.96*E* + 01	5.30*E* + 00	6.75*E* + 01	6.54*E* + 00	9.42*E* + 01	9.66*E* + 00	1.28*E* + 02	9.67*E* + 00	1.58*E* + 02	1.26*E* + 01
*F*11	2.60*E* + 01	2.28*E* + 00	**2.58*E* + 01**	**1.25*E* + 00**	2.61*E* + 01	1.39*E* + 00	2.91*E* + 01	1.67*E* + 00	3.06*E* + 01	1.54*E* + 00	3.21*E* + 01	9.02*E* − 01	3.36*E* + 01	1.22*E* + 00
*F*12	**4.80*E* + 03**	**4.26*E* + 03**	5.80*E* + 03	4.27*E* + 03	1.19*E* + 04	7.50*E* + 03	2.01*E* + 04	7.36*E* + 03	3.60*E* + 04	9.39*E* + 03	5.00*E* + 04	1.09*E* + 04	6.02*E* + 04	1.26*E* + 04
*F*13	**1.22*E* + 00**	**1.05*E* − 01**	1.45*E* + 00	1.15*E* − 01	2.21*E* + 00	1.68*E* − 01	3.09*E* + 00	2.69*E* − 01	4.03*E* + 00	2.26*E* − 01	4.76*E* + 00	2.87*E* − 01	5.29*E* + 00	4.37*E* − 01
*F*14	1.24*E* + 01	3.26*E* − 01	1.24*E* + 01	3.44*E* − 01	**1.23*E* + 01**	**2.41*E* − 01**	1.28*E* + 01	2.41*E* − 01	1.31*E* + 01	1.75*E* − 01	1.33*E* + 01	1.39*E* − 01	1.33*E* + 01	1.79*E* − 01

Table 2: Comparison between JADE with proposed population adaptation and JADE with different population sizes.

Fcn	JADE with different population sizes		JADE with adaptive population	
	Mean	Std.	Mean	Std.
$F1$ ($NP = 50, 100, 200$ & 400)	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$
$F2$ ($NP = 100$)	$1.79E-28$	$1.50E-28$	$7.30E-28$	$1.14E-28$
$F3$ ($NP = 400$)	$9.52E-08$	$4.15E-07$	**$5.89E-13$**	**$2.73E-12$**
$F4$ ($NP = 400$)	$5.64E-18$	$1.68E-17$	$3.56E-18$	$5.09E-18$
$F5$ ($NP = 400$)	$2.91E+01$	$2.55E+01$	$2.86E+01$	$2.80E+01$
$F6$ ($NP = 50$)	$4.06E+00$	$1.47E+01$	$3.08E+00$	$1.29E+01$
$F7$ ($NP = 1000$)	**$4.22E-04$**	**$2.64E-04$**	$1.23E-03$	$3.22E-03$
$F8$ ($NP = 100$)	$2.08E+01$	$3.25E-01$	$2.08E+01$	$3.17E-01$
$F9$ ($NP = 50$ & 100)	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$
$F10$ ($NP = 100$)	$2.58E+01$	$5.13E+00$	**$2.20E+01$**	**$4.39E+00$**
$F11$ ($NP = 100$)	$2.58E+01$	$1.25E+00$	$2.50E+01$	$1.20E+00$
$F12$ ($NP = 50$)	$4.80E+03$	$4.26E+03$	**$3.92E+03$**	**$1.80E+03$**
$F13$ ($NP = 50$)	$1.22E+00$	$1.05E-01$	$1.30E+00$	$1.18E-01$
$F14$ ($NP = 200$)	$1.23E+01$	$2.41E-01$	$1.23E+01$	$2.30E-01$

The maximum number of function evaluations considered is $10000 * D$ (100000 for 10D; 300000 for 30D; 500000 for 50D; 1000000 for 100D). The algorithm is terminated when the maximum number of function evaluations is reached.

In the current work, each algorithm is run 30 times independently on each problem. To compare the performance of different algorithms, we employ two types of statistical tests, namely t-test and Wilcoxon rank-sum test. The t-test being a parametric method can be used to compare the performance of two algorithms on a single problem. To compare the performance of two algorithms over a set of different problems, we use a nonparametric test such as the Wilcoxon rank-sum test.

The proposed algorithm employs "DE/current-to-$p$best" mutation strategy along with the binomial crossover. As mentioned above, the proposed adaptation scheme works in the bidimensional parametric space (F and CR). In the proposed algorithm, we initially sample 20D solution vectors, out of which 100 individuals are selected as the population members at the start of every generation. After the generation the solution vectors are replaced. In addition to the parameters of the DE algorithm, the parameters of the GaA algorithm are set to the same values that are proposed in [10].

4.1. Effect of Population Size and Population Adaptation in DE Algorithm. At first, we demonstrate the effect of population size (NP) on the performance of the adaptive DE algorithms. To demonstrate the effect, we consider the JADE algorithm as the base algorithm and evaluated the 30D versions of 14 benchmark problems of CEC 2005 with different population sizes such as 50, 100, 200, 400, 600, 800, and 1000. The results are summarized in Table 1.

In Table 1, corresponding to each problem, statistically significant results (based on the statistical t-test) are marked in bold font. From the results, it can be observed that JADE with a single population size setting is not apt on all benchmark problems. JADE with NP values less than 400 perform better on almost all the benchmark problems except $F7$. In addition, $NP = 50$ and $NP = 100$ perform equally better on most of the benchmark problems compared to $NP = 200$ and $NP = 400$, in terms of solution quality. However, there $NP = 400$ shows a significantly better performance compared to $NP = 50$ and $NP = 100$ on $F3$, $F4$, and $F5$. During the experimentation, a similar observation was made with the benchmark problems with dimensionality 10D, 50D, and 100D. Therefore, from the results it can be observed that a single setting of the population even when the strategy parameters (F and CR) are self-adapted may not result in optimal performance on a given set of benchmark problems. In other words, these results demonstrate the need for the self-adaptation of the population in DE algorithm.

In Table 2, we compare the performance of the proposed population adaptation scheme with JADE using fixed population size. In Table 2, the best results corresponding to each benchmark problem obtained using JADE with fixed population size is present in Table 2. In Table 2, the population size in the first column indicates the population size at which the results (reported in columns 2 and 3 are taken from Table 1) are taken. The results in columns 4 and 5 correspond to mean and standard deviation values obtained using JADE with the proposed population adaptation scheme.

From the results in Table 2, it can be observed that the proposed population adaptation scheme is slightly better or equal (based on t-test) to the best results obtained using fixed population size. In addition, the adaptation of population alleviates the need to tune the parameter depending on the characteristics of the benchmark problems. In other words, the proposed population adaptation scheme is robust and can overcome the need to tune the population size parameter using trial-and-error method which is time consuming. In addition, Figure 1 shows the convergence characteristic of JADE with fixed population of 400 and JADE with the proposed adaptive population techniques. From the figure, it is clear that the proposed population adaptation can better balance between the exploration and exploitation capabilities. In addition, the Wilcoxon rank-sum test results demonstrate

Table 3: Comparison between JADE with adaptive population and proposed algorithm.

	10D				30D				50D				100D			
Fcn	JADE with Adaptive *NP*		Proposed Algorithm		JADE with Adaptive *NP*		Proposed Algorithm		JADE with Adaptive *NP*		Proposed Algorithm		JADE with Adaptive *NP*		Proposed Algorithm	
	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
*F*1	4.25E − 23	1.77E − 22	**0.00E + 00**	**0.00E + 00**	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	2.80E − 22	1.28E − 21	**2.36E − 30**	**7.14E − 30**	4.52E − 20	1.05E − 19	**5.44E − 27**	**2.98E − 28**
*F*2	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	7.30E − 28	1.14E − 28	1.69E − 28	3.42E − 28	1.35E + 03	7.41E + 03	**2.44E − 17**	**6.19E − 17**	2.32E + 04	7.09E + 04	**2.71E − 03**	**2.19E − 03**
*F*3	5.51E − 25	2.81E − 25	1.16E − 25	2.78E − 26	5.89E − 13	2.73E − 12	**7.21E − 18**	**2.41E − 17**	7.16E + 02	1.42E + 03	3.89E + 02	3.17E + 02	4.64E + 05	1.42E + 05	2.27E + 05	9.69E + 04
*F*4	6.02E − 29	1.01E − 28	0.00E + 00	0.00E + 00	3.56E − 18	5.09E − 18	**7.42E − 23**	**1.42E − 22**	4.08E + 03	1.55E + 04	**1.62E − 02**	**7.61E − 02**	1.03E + 04	5.51E + 04	**1.17E + 03**	**8.51E + 02**
*F*5	3.75E − 11	3.30E − 11	**4.32E − 15**	**3.58E − 16**	2.86E + 01	2.80E + 01	**3.43E + 00**	**5.61E + 00**	1.10E + 03	3.30E + 02	1.03E + 03	2.54E + 02	3.20E + 03	9.31E + 02	1.13E + 03	5.21E + 02
*F*6	4.02E + 00	2.28E + 00	**1.73E − 27**	**6.72E − 27**	3.08E + 00	1.29E + 01	**1.81E − 01**	**7.51E − 01**	7.87E + 01	8.35E + 01	**5.40E + 00**	**8.04E + 00**	**3.94E + 01**	**2.62E + 01**	4.20E + 01	3.48E + 01
*F*7	4.57E − 02	1.72E − 02	**9.57E − 03**	**8.08E − 03**	1.23E − 03	3.22E − 03	**0.00E + 00**	**0.00E + 00**	2.05E − 03	7.07E − 03	**3.29E − 04**	**1.80E − 03**	1.40E − 03	4.37E − 03	2.79E − 03	4.69E − 03
*F*8	2.04E + 01	6.93E − 02	2.01E + 01	4.21E − 02	2.08E + 01	3.17E − 01	2.02E + 01	2.02E − 01	2.09E + 01	4.58E − 01	2.03E + 01	2.33E − 01	2.09E + 01	5.59E − 01	2.04E + 01	3.01E − 01
*F*9	3.31E − 13	1.76E − 13	**0.00E + 00**	**0.00E + 00**	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	0.00E + 00	1.33E − 01	7.27E − 01
*F*10	5.98E + 00	1.61E + 00	**3.26E + 00**	**8.98E − 01**	2.20E + 01	4.39E + 00	**1.47E + 01**	**2.83E + 00**	3.82E + 01	7.15E + 00	**3.46E + 01**	**1.32E + 01**	2.53E + 02	1.29E + 02	**1.20E + 02**	**2.26E + 01**
*F*11	5.01E + 00	5.72E − 01	5.42E + 00	6.25E − 01	2.50E + 01	1.20E + 00	**2.15E + 01**	**8.05E + 00**	5.10E + 01	1.96E + 00	**4.35E + 01**	**7.45E + 00**	2.40E + 02	1.52E + 02	**1.26E + 02**	**3.09E + 00**
*F*12	9.08E + 01	7.62E + 01	**6.28E + 01**	**2.85E + 02**	3.92E + 03	1.80E + 03	**1.85E + 03**	**3.14E + 03**	2.20E + 05	1.57E + 05	**6.85E + 03**	**8.10E + 03**	1.57E + 06	1.76E + 06	**3.90E + 04**	**2.89E + 04**
*F*13	4.83E − 01	4.95E − 02	**3.73E − 01**	**5.14E − 01**	1.30E + 00	1.18E − 01	1.26E + 00	1.51E − 01	3.47E + 00	2.80E − 01	2.97E + 00	3.45E − 01	7.85E + 00	1.02E + 00	7.52E + 00	1.32E + 00
*F*14	2.85E + 00	2.97E − 01	2.82E + 00	3.33E − 01	1.23E + 01	2.30E − 01	1.24E + 01	3.13E − 01	2.16E + 01	3.53E − 01	2.18E + 01	3.98E − 01	4.57E + 01	4.28E − 01	3.56E + 01	5.23E − 01
Wilcoxon Ranksum Test			**1**				**1**				**1**				**1**	

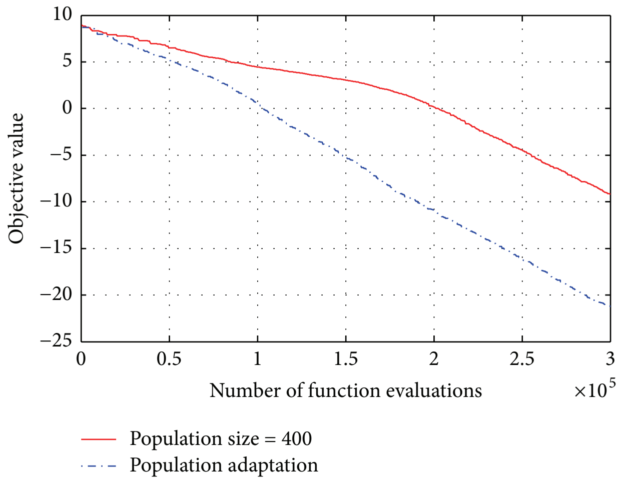

Figure 1: Convergence characteristics of JADE with fixed and adaptive population for $F3$.

that the adaptive population is better than each of the individual population sizes (50, 100, 200, 400, 600, 800, and 1000).

4.2. Adaptation of F and CR Using Gaussian Adaptation. In this section, we evaluate the performance of the proposed algorithm which includes the population adaptation and adapts the strategy parameters (F and CR) using Gaussian adaptation. The proposed algorithm is evaluated using 10D, 30D, 50D, and 100D versions of the 14 benchmark problems of CEC 2005 and is compared with the JADE with population adaptation. The results are summarized in Table 3 and statistically significant (based on t-test) results are marked by bold font.

From the results in Table 3, it can be observed that the proposed algorithm is always better or equal but not worse, statistically in terms of the solution quality. The significant difference in the performance between JADE and the proposed GaA based parameter adaptation is in the linked problems such as $F3$ and $F10$. In these problems, the superiority of the proposed algorithm can be attributed to the modelling of the interdependency between the two strategy parameters (F and CR). Even in the unimodal problems such as $F1$ and $F2$, the difference in the performance becomes significant as dimensionality increases.

In Table 3, the last column contains the results of Wilcoxon rank-sum test results. Number 1 below the proposed algorithm indicates that the proposed algorithm is statistically better than JADE with adaptive NP alone.

5. Conclusion

In this paper, we propose a DE algorithm where the population is adapted in addition to the strategy parameters. Unlike in most of the adaptive DE algorithms, the proposed algorithm adapts the DE strategy parameters by taking into account the interactions between them. The interactions are modelled using a Gaussian distribution in the bivariate parameter space, referred to as Gaussian adaptation. With the help of simulations, we were able to demonstrate the superiority of the proposed population adaptation technique compared to the fixed population size in terms of balancing the exploitation and explorations stages of the evolution. In addition, the Gaussian Adaptation of the strategy parameters demonstrates the importance of considering the interactions between the parameters while designing parameter adaptation techniques in evolutionary algorithms.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This study was supported by the BK21 Plus project funded by the Ministry of Education, Republic of Korea (21A20131600011).

References

[1] R. Mallipeddi, J. P. Lie, S. G. Razul, P. N. Suganthan, and C. M. S. See, “Robust adaptive beamforming based on co-variance matrix reconstruction for look direction mismatch,” *Progress in Electromagnetics Research Letters*, vol. 25, pp. 37–46, 2011.

[2] M. K. Venu, R. Mallipeddi, and P. N. Suganthan, “Fiber Bragg grating sensor array interrogation using differential evolution,” *Optoelectronics and Advanced Materials, Rapid Communications*, vol. 2, no. 11, pp. 682–685, 2008.

[3] R. Mallipeddi, S. Jeyadevi, P. N. Suganthan, and S. Baskar, “Efficient constraint handling for optimal reactive power dispatch problems,” *Swarm and Evolutionary Computation*, vol. 5, pp. 28–36, 2012.

[4] R. Gämperle, S. D. Müller, and P. Koumoutsakos, “A parameter study for differential evolution,” in *The Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*, 2002.

[5] Z. Jingqiao and A. C. Sanderson, “An approximate gaussian model of differential evolution with spherical fitness functions,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’07)*, pp. 2220–2228, 2007.

[6] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, “Differential evolution algorithm with ensemble of parameters and mutation strategies,” *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 1679–1696, 2011.

[7] V. L. Huang, A. K. Qin, and P. N. Suganthan, “Self-adaptive differential evolution algorithm for constrained real-parameter optimization,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’06)*, pp. 17–24, July 2006.

[8] J. Zhang and A. C. Sanderson, “JADE: adaptive differential evolution with optional external archive,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[9] J. Teo, “Exploring dynamic self-adaptive populations in differential evolution,” *Soft Computing*, vol. 10, no. 8, pp. 673–686, 2006.

[10] C. Müller and I. Sbalzarini, “Gaussian adaptation revisited—an entropic view on covariance matrix adaptation,” in *Applications of Evolutionary Computation*, C. Chio, S. Cagnoni, C. Cotta et al., Eds., vol. 6024, pp. 432–441, Springer, Berlin, Germany, 2010.

[11] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[12] R. Storn, "On the usage of differential evolution for function optimization," in *Proceedings of the Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS '96)*, pp. 519–523, Berkeley, Calif, USA, June 1996.

[13] A. Iorio and X. Li, "Solving rotated multi-objective optimization problems using differential evolution," in *Proceedings of the Australian Conference on Artificial Intelligence*, Cairns, Australia, 2004.

[14] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[15] R. Mallipeddi, "Harmony search based parameter ensemble adaptation for differential evolution," *Journal of Applied Mathematics*, vol. 2013, Article ID 750819, 12 pages, 2013.

[16] M. G. H. Omran, A. Salman, and A. P. Engelbrecht, "Self-adaptive differential evolution," in *Computational Intelligence and Security*, vol. 3801 of *Lecture Notes in Computer Science*, pp. 192–199, Springer, Berlin, Germany, 2005.

[17] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[18] S. Das, A. Konar, and U. K. Chakraborty, "Two improved differential evolution schemes for faster global search," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '05)*, pp. 991–998, June 2005.

[19] H. A. Abbass, "The self-adaptive Pareto differential evolution algorithm," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, vol. 1, pp. 831–836, IEEE, Honolulu, Hawaii, USA, May 2002.

[20] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Computing*, vol. 9, no. 6, pp. 448–462, 2005.

[21] D. Zaharie, "Control of population diversity and adaptation in differential evolution algorithms," in *Proceedings of the 9th International Conference on Soft Computing*, pp. 41–46, Brno, Czech Republic, 2003.

[22] D. Zaharie and D. Petcu, "Adaptive pareto differential evolution and its parallelization," in *Proceedings of the 5th International Conference on Parallel Processing and Applied Mathematics*, pp. 261–268, Czestochowa, Poland, 2003.

[23] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, pp. 1769–1776, September 2005.

[24] B. Yuan and M. Gallagher, "Experimental results for the special session on real-parameter optimization at CEC 2005: a simple, continuous EDA," in *Proceedings of the IEEE Congress on Evolutionary Computation (IEEE CEC '05)*, pp. 1792–1799, September 2005.

[25] J. J. Liang, P. N. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '05)*, pp. 71–78, Pasadena, Calif, USA, June 2005.