Hindawi
Complexity
Volume 2021, Article ID 8878780, 12 pages
https://doi.org/10.1155/2021/8878780

WILEY | Hindawi

Research Article

Large-Scale Evolutionary Strategy Based on Gradient Approximation

Jin Jin [1,2]

[1]*Chengdu Institution of Computer Application, Chengdu 610041, China*
[2]*University of Chinese Academy of Sciences, Beijing 100049, China*

Correspondence should be addressed to Jin Jin; jinjin@nsu.edu.cn

Received 4 September 2020; Revised 5 April 2021; Accepted 3 May 2021; Published 17 May 2021

Academic Editor: Guang Li

For large-scale optimization, CMA-ES has the disadvantages of high complexity and premature stagnation. An improved CMA-ES algorithm called GI-ES was proposed in this paper. For the problem of high complexity, the method in this paper replaces the calculation of a covariance matrix with the modeling of expected fitting degrees for a given covariance matrix. At the same time, to solve the problem of premature stagnation, this paper replaces the historical information of elite individuals with the historical information of all individuals. The information can be seen as approximate gradients. The parameters of the next generation of individuals are generated based on the approximate gradients. The experimental results were tested using CEC 2010 and CEC2013 LSGO benchmark test suite, and the experimental results verified the effectiveness of the algorithm on a number of different tasks.

1. Introduction

Most machine learning problems can be modeled as optimization problems, and as the amount of data increases, the model becomes more and more complex. Therefore, the problem of large-scale optimization has become the focus of most researchers. The covariance matrix adaptation evolutionary computation (CMA-ES) [1] is one of the most powerful evolutionary strategies for global optimization. It is an algorithm based on the population probability distribution, very similar to the estimation of distribution algorithm (EDA) [2]. A common shortcoming of simple evolutionary strategies is that the noise parameters of the standard deviation are fixed. CMA-ES can automatically adjust the standard deviation according to the distribution of the population, which brings two benefits:

(1) Increasing the diversity of the population, so that the algorithm can jump out of the local optimal solution

(2) Adjusting the standard deviation parameters so that the algorithm can adaptively adapt to the fitness terrain

In addition, because CMA-ES can use the information of the optimal solution to adjust its parameters at the same time, it can expand the search scope when the optimal solution is far away or narrow the search scope when the optimal solution is near. Due to these advantages, CMA-ES algorithm, as one of the most popular gradient-free optimization algorithms, has become the choice of many researchers and practitioners.

Despite its huge advantage in solving optimization problems, CMA-ES suffers some limitations when dealing with LSOPs:

Time-consuming: because the basic operation of CMA-ES is based on covariance, and each generation takes $O(n^2)$ time and $O(n^2)$ space, sampling the new population requires additional $O(n^3)$ calculations to decompose the matrix. CMA-ES relies on spectral decomposition when dealing with numerical errors and ill-conditioned conditions, which is generally considered to be computationally inefficient compared to other decomposition techniques.

Lack of diversity in population: another obvious disadvantage is that CMA-ES assessed some of the best

individuals. Although it can speed up convergence to some extent, this strategy discards most of the information. This prevents the CMA-ES from performing well when dealing with ill-conditional problems. We all know that great people in life have certain qualities that we can learn from, but some people who fail also keep a record of "not doing" something. It is important for the next generation to make better calculations and evaluations.

These two limitations may prevent the use of CMA-ES for LSGOs.

To solve the above problems, this paper proposes an evolutionary strategy based on gradient information utilization (GI-ES), which extends the application of CMA-ES in the field of large-scale optimization. The characteristics of GI-ES are as follows:

(1) The fitness scores were optimized for each sampling scheme for each GI-ES. The best-performing scheme is likely to perform better in the sampled generation if the expected results are good enough. Maximizing the expected fit score of a sampling scheme is the same as maximizing the overall fit score of the sample in a given sample.

(2) The gradient information is simulated using individual information, and the approximate gradient information is used to guide the search.

(3) GI-ES uses an approximate gradient for the search directions, allowing the algorithm to adapt to the fitness landscape, on which the variables depend. This process generates a fitness score evaluation based on the expected fitting score. The gradient signal is obtained by using the maximum likelihood estimation method of the model described above. It differs from traditional evolutionary computation in that it represents the "population" as a parameterized distribution. It uses a search gradient to update the parameters of the distribution, which can be calculated using the adaptive values of historical data.

Following the introduction section, we first discuss the related work of large-scale optimization and the utilization of historical information strategies for evolutionary computation in Section 2. In Section 3, the background and motivation of this paper are introduced. Section 4 describes the detailed implementations of GI-ES. Thereafter, the simulation results on the benchmark test suites are examined to evaluate the effectiveness of the proposed approach in Section 5. Finally, Section 6 summarizes this paper.

2. Related Work

In recent years, large-scale optimization has become a hot research topic, and many large-scale benchmark functions have been put forward to examine the merits of large-scale optimization algorithms. Researchers have done a lot of useful work and proposed many solutions to solving large-scale global optimization problems.

There are currently two main research directions for large-scale optimization problems:

(1) Decomposition-based algorithm: the dimensionality reduction (decomposition) is carried out for large-scale problems in the form of grouping, so as to decompose the large-scale problems into multiple sub-problems to solve. Multiple sub-problems are optimized using evolutionary algorithms within the framework of cooperative coevolution (CC).

(2) Instead of directly decomposing large-scale problems, optimization is carried out by combining multiple local search algorithms, each with its own set of parameters.

The rest of this section will further discuss some algorithms in the recent CEC large-scale optimization competition.

CEC2008 is the first known LSGO competition. Multiple trajectory search (MTS) [3] was the first winner. Then, MTS-LS1, MTS-LS2, and MTS-LS3 are the improved version of MTS. Some DE-based algorithms have achieved good results in LSGO competitions. Self-adaptive differential evolution with multi-trajectory search (SaDE-MMTS) [4] is a hybridization algorithm, which integrates JADE [5] and modified MTS-LS1. MA-SW-Chains [6] is an extension of MA-CMA-Chains, where CMA was replaced with Solis Wets (SW).

In the CEC2010 competition, MA-SW-Chains was the winner. Ensemble Optimization Evolutionary Algorithm (EOEA) [7] was the second-ranked algorithm. In EOEA, the optimization process is divided into two stages, namely, global shrinking and local exploration. In EOEA, EDA based on the Mixed Gaussian and Cauchy Model (MUEDA) is used to achieve more quickly convergence. The goal of the second stage is exploitation. The third place in CEC2010 is the Differential Ant Colony Algorithm (DASA) [8]. It tries to solve LSGO by converting the real parameter optimization problem into a graph search problem.

Improved multiple offspring sampling (MOS) [9] was the winner of CEC2012. MOS combines SW and MTS-LS as two local searches. Self-adaptive differential evolution algorithm (jDElsgo) was proposed in [10]. After continuous improvement, jDElsgo was ranked second in CEC2012. The third rank was cooperative coevolution evolutionary algorithm with global search (CCGS). CCGS is considered an extension of EOEA [11]. Cooperative coevolution with delta grouping (DECC-DML) was proposed in [12] to enhance the performance of CC framework on non-separable problems.

CEC2013 uses a new benchmark functions. The winner of CEC2013 competition was modified MOS [13]. DECC-G [14] was the reference algorithm in CEC2013. The second-ranked algorithm was smoothing and auxiliary function-based cooperative coevolution for global optimization (SACC) [15]. SACC adopted parallel search for the first time under the CC framework.

Bi-space interactive cooperative coevolutionary algorithm (BICA) [16] is a two-space co-evolutionary algorithm framework that evolves in two spaces. The model

evolves to provide better grouping, and the individual evolves to achieve better adaptability. SHADE with Iterative Local Search (SHADE-ILS) [17] iteratively combines the modern differential evolution algorithm with a local search method selected from various search methods. The selection of local search methods is dynamic and takes into account the improvements they have made in the previous enhancement phase to determine the best method for the problem in each situation. In LSHADE-SPA [18], differential evolution with linear population size reduction is used for global exploration, while a modified version of multitrack search is used for local exploitation.

A hybrid adaptive evolutionary differential evolution (HACC-D) was also proposed in CEC2014 [19]. HACC-D belongs to the CC class of algorithms. JADE and SaNSDE are used as CC subcomponent optimization algorithms. Scaling up Covariance Matrix Adaptation Evolution Strategy using cooperative coevolution (CC-CMA-ES) is another CC-based competitive algorithm. The basic optimizer in CC-CMA-ES is CMA-ES.

For a detailed overview of the state-of-the-art large-scale optimization algorithms, please refer to [20]. This paper proposes an algorithm for large-scale optimization problems, called GI-ES. The algorithm has been verified on the CEC2010 and CEC2013 test sets. The experimental results verify the potential value of the algorithm.

3. Background and Motivation

This section briefly introduces the background of CMA-ES algorithm. With that in mind, the paper focuses on the utilization of information by the CMA-ES algorithm. Based on the analysis of different information utilization methods, this paper proposes a model of gradient information utilization.

3.1. CMA-ES. The CMA-ES algorithm is an evolutionary strategy algorithm proposed by Hansen et al. [1]. CMA-ES algorithm uses a Gaussian distribution to sample the solution space of the optimization problem. The parameters of the distribution are updated according to a sample selection mechanism. Iterate through the update process until all the conditions are met.

For the objective function $\operatorname{Arg\,min} f(x)$, CMA-ES generates a new generation of population by estimating the distribution of the objective function; that is, the points in the new population are obtained from the normal distribution $N(m, \sigma C)$, where $m \in R^n$ represents the mean value, $C \in R^{n\times n}$ is a positive definite matrix, and this matrix is called the covariance matrix. In the algorithm, the covariance matrix C is continuously adjusted to make the distribution of the search points closer to the equipotential line of the target function; that is, the ideal covariance matrix should be the inverse matrix equal to the hessian matrix, though this is difficult to achieve for more complex functions. Next, we briefly introduce the basic ideas behind CMA-ES.

3.1.1. Sampling. For a given objective function, CMA-ES first assumes the existence of an optimal fitness terrain to make the search move in the optimal direction. For convex quadratic function, the correlation between variables is linear and can be completely eliminated. So CMA-ES treats these functions "as" spherical functions and effectively solves them. This strategy is also applicable to general black-box objective functions, because their local landscapes can be approximated as convex quadratic functions. In general, the optimal covariance is not known. CMA-ES sampling through the multivariate Gaussian distribution $N(m, \sigma C)$. σ is the search step size, which is used to control the local search capability of Gaussian distribution. The number of samples λ is used for each sample, and then the fitness value of the samples is calculated. In each generation, CMA-ES provides a multivariate normally distributed parameter for sampling the next generation.

$$x_k^{g+1} \sim m^g + o^g N\left(0, C^g\right), \quad k = 1, 2, \ldots, \lambda. \tag{1}$$

3.1.2. Update. The CMA-ES algorithm updates the parameters (m, C, and σ). Update operations are more complex. The new mean $m^{(g+1)}$ was updated with the μ excellent individuals obtained in the g th generation. The contribution of individuals to the mean value is considered by inertia weighting.

$$m^{g+1} = \sum_{i=1}^{\mu} w_i x_{i\lambda}^{g+1}, \tag{2}$$

where

$$\sum_{i=1}^{\mu} w_i = 1, \quad w_1 \geq w_2 \geq \cdots \geq w_\mu \geq 0, \tag{3}$$

where $\mathbf{x}_{\mathbf{i},\lambda}^{\mathbf{g+1}}$ represents μ individuals with the highest fitness ranking among λ samples. Because the weight of each sample is different, the weight of the sample with better fitness is bigger.

There are two ways to update covariance: rank − 1 and rank − μ. This paper adopts rank − 1 update strategy. The update strategy of covariance matrix is as follows:

$$\begin{cases} C^{(0)} = I_n, \\ C^{(g+1)} = (1-c)C^{(g)} + cp^{(g)}\left(p^{(g)}\right)^T, \end{cases} \tag{4}$$

where I_n is the n-dimensional integer matrix, c is the learning rate, the learning rate range is [0, 1], and p^g is the evolutionary path, that is, the memory of mean deviation will decay along with the optimization process. Therefore, CMA-ES algorithm is used to model the variance matrix of multivariate Gaussian distribution, instead of using maximum likelihood estimation to model the sample. In this way, the successful step size update in the previous step will be highly likely to appear again in the next generation. For the detailed method of updating, see [21].

The step size is

$$\sigma^{g+1} = \sigma^{g} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\left\|\mathbf{p}_\sigma^{g+1}\right\|}{E\|\mathbf{N}(0,\mathbf{I})\|} - 1\right)\right), \tag{5}$$

among which

$$E\|N(0,I)\| = \frac{\sqrt{2}T(D+1/2)}{\Gamma(D/2)} \approx \sqrt{D} + \mathcal{O}\left(\frac{1}{n}\right). \tag{6}$$

D is the evolutionary path.

3.2. Motivation. Because CMA-ES can use the information provided by the optimal solution to adjust its mean value and variance simultaneously, it can expand the search scope when the optimal solution is far away, or narrow the search scope when the optimal solution is near. But the covariance matrix takes $O(n^2)$ in space and time. Throughout the iteration process, only the individuals with higher fitness ranking are considered. When faced with some simple problems, the algorithm can quickly converge, but when faced with large-scale optimization problems, this can quickly lead to premature stagnation.

Survival of the fittest is an important part of evolutionary computing, but the diversity of populations is crucial to the performance of the algorithm. Through evolution, the traditional CMA-ES preserves the optimal individuals and influences the distribution of the next generation by learning the individuals of the optimal parts. That is, in the evolutionary past, populations tended more and more toward "elite" individuals than toward "bad" individuals. But the bad ones retain key information about what not to do. Success studies are everywhere, but what is really useful is often the experience of unsuccessful people, because it contains long-term life observations and lessons. These constitute the motivation for writing this article.

4. Proposed Approach

The proposed algorithm GI-ES adopts the basic framework of CMA-ES but makes some improvements to CMA-ES. In the proposed scheme, keep all the information about each scheme available to each generation, good or bad. With these gradient signal assessments, we can move the whole scheme in a better direction for the next generation. Since we need to evaluate the gradient, we can use the standard stochastic gradient descent algorithm (SGD) applied to deep learning [22].

4.1. GI-ES. The fitness score was optimized for each sampling scheme in GI-ES. If the expected results are good enough, the best-performing scheme in the sampling generation may perform better. Maximization of the expected fitness score of a sampling scheme is actually equivalent to maximization of the overall fitness score.

Assuming that $\mathbf{z}$ is the sampling scheme vector of the probability distribution function $\pi(\mathbf{z}, \theta)$, we can define the expected value of the objective function F as

$$J(\theta) = E_\theta[F(\mathbf{z})] = \int F(\mathbf{z})\pi(\mathbf{z},\theta)\mathrm{d}\mathbf{z}, \tag{7}$$

where θ represents the parameter of the probability distribution function. For example, if π is a normal distribution, θ is μ and σ. For our simple two-dimensional problem, every whole $\mathbf{z}$ is a two-dimensional vector (x, y). Using the same logarithm likelihood method as in REINFORCE, we can calculate the gradient of $J(\theta)$:

$$\nabla_\theta J(\theta) = E_\theta\left[F(\mathbf{z})\nabla_\theta \log\, \pi(\mathbf{z},\theta)\right]. \tag{8}$$

In a sample of size N, we have scheme $\mathbf{z}^1, \mathbf{z}^2, \ldots, \mathbf{z}^N$, so that the gradient can be estimated by summation:

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N} F\left(\mathbf{z}^i\right)\nabla_\theta \log\, \pi\left(\mathbf{z}^i,\theta\right). \tag{9}$$

With the above gradient, we can use a learning rate of alpha (say 0.01) and start our theta optimization of the probability distribution function so that our sampling scheme gets a higher fitness score on the target function F. Using SGD or Adam [23] algorithm, we can update θ in the next generation:

$$\theta \longrightarrow \theta + \alpha\nabla_\theta J(\theta). \tag{10}$$

After the probability distribution function is updated, the new competitive scheme z can be sampled until an appropriate solution is obtained. Since relevant parameters are not used, the efficiency of this algorithm is $O(N)$.

GI-ES adopts an approximate gradient as the direction of search. It represents the "population" in traditional evolutionary computation as a parameterized distribution $\pi(\mathbf{z}, \theta)$.

4.1.1. Multinormal Distribution. In this proposed method, multivariate normal distribution is used as the parameterized distribution. The parameter θ of multivariate normal distribution is (μ, Σ). $\mu \in \mathbb{R}^d$ is mean of multivariate normal distribution, and $\Sigma \in \mathbb{R}^{d\times d}$ is the covariance matrix. To sample more efficiently, we need a matrix $\mathbf{A} \in \mathbb{R}^{d\times d}$, meeting $\mathbf{A}^\top\mathbf{A} = \Sigma$. Then, the problem can be simplified to a standard multivariate normal distribution problem. $\pi(\mathbf{z}|\theta)$ represents the probability density function of the multinormal distribution.

$$\begin{aligned}\pi(\mathbf{z}|\theta) &= \frac{1}{(\sqrt{2\pi})^d|\det(\mathbf{A})|}\cdot\ \exp\left(-\frac{1}{2}\left\|\mathbf{A}^{-1}\cdot(\mathbf{z}-\boldsymbol{\mu})\right\|^2\right)\\ &= \frac{1}{\sqrt{(2\pi)^d\det(\boldsymbol{\Sigma})}}\cdot\ \exp\left(-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu})^\top\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\right).\end{aligned} \tag{11}$$

To calculate the gradient information of the multivariate Gaussian variable, the logarithm of the probability density is obtained, so that the gradient can be estimated by summation:

$$\log \pi\pi(\mathbf{z}|\theta) = -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log \det\boldsymbol{\Sigma} - \frac{1}{2}(\mathbf{z}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu}), \tag{12}$$

so $\nabla_{\mu}\log \pi(\mathbf{z}|\theta)$ and $\nabla_{\Sigma}\log \pi(\mathbf{z}|\theta)$ can be obtained in (15) and (14).

$$\nabla_{\Sigma}\log \pi(\mathbf{z}|\theta), \tag{13}$$

$$\nabla_{\boldsymbol{\Sigma}}\log \pi(\mathbf{z}|\theta) = \frac{1}{2}\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})(\mathbf{z}-\boldsymbol{\mu})^{\top}\boldsymbol{\Sigma}^{-1} - \frac{1}{2}\boldsymbol{\Sigma}^{-1}. \tag{14}$$

Then, update the parameters with the calculated gradient information.

$$\theta \leftarrow \theta + \eta\nabla_{\theta}J. \tag{15}$$

To adapt to exploring or mining solution space, the algorithm can change the solution distribution according to the parameters of the solution that it is exploring.

4.1.2. The Technique of GI-ES. Further, A can be decomposed into a scale parameter σ, and a normalized covariance factor B satisfying $\det(B) = 1$. This decoupling form of two orthogonal components can be independently learned.

The advantage of overall information utilization is to prevent information loss, but outliers still need to be considered. Therefore, this paper adopts the method of fitting degree shaping to solve outliers dominant situation [24]. In this method, according to the fitness value information, the population individuals are ranked according to the fitness from small to large. Calculate the utility value according to the fitness value $u_1 \geq \cdots \geq u_{\lambda}$.

$$u_k = \frac{\max(0,\ \log((\lambda/2)+1) - \log(k))}{\sum_{j=1}^{\lambda}\max(0,\ \log((\lambda/2)+1) - \log(j))} - \frac{1}{\lambda}. \tag{16}$$

The complexity of each covariance matrix update is $O(d^3)$. The complexity can be reduced to $O(d^2)$ by calculating the update of local non-exponential coordinates. In this case, the update of gradient information can be decomposed into the following components:

$$\nabla_{\boldsymbol{\delta}}J \leftarrow \sum_{k=1}^{\lambda} u_k \cdot \mathbf{s}_k, \tag{17}$$

$$\nabla_{\mathbf{M}}J \leftarrow \sum_{k=1}^{\lambda} u_k \cdot (\mathbf{s}_k\mathbf{s}_k^{\top} - \mathrm{II}), \tag{18}$$

$$\nabla_{\sigma}J \leftarrow \frac{\mathrm{tr}(\nabla_{\mathbf{M}}J)}{d}, \tag{19}$$

$$\nabla_{\mathbf{B}}J \leftarrow \nabla_{\mathbf{M}}J - \nabla_{\sigma}J \cdot \mathrm{II}. \tag{20}$$

The pseudocode of GI-ES is given in Algorithm 1.

5. Experiments and Analysis

In this section, GI-ES is used to compare with the state-of-the-art algorithms to verify the effectiveness of the proposed algorithm. Three performance analysis experiments were performed. First, GI-ES was evaluated using CEC2010. Second, CEC2013 is used to evaluate the GI-ES and compared with nine state-of-the-art algorithms. Finally, a parametric analysis was performed to study the effect of each component in GI-ES.

The first experiment is CEC2010, the second experiment is CEC2013, and the third experiment is statistical analysis. The benchmark functions are shown as follows. The dimensions (D) of all functions are 1000 except for two overlapping functions, F13 and F14 in CEC2013, where D is 905.

(1) The CEC2010 contains 20 test questions. These test functions can be divided into four classes:

- (i) F1–F3: separable functions
- (ii) F4–F8: partially separable functions, in which a small number of variables are dependent, while all the remaining ones are independent ($m = 50$)
- (iii) F9–F18: partially separable function that consists of multiple independent subcomponents, each of which is m-non-separable ($m = 50$)
- (iv) F19–F20: fully non-separable functions

For more detailed features, please refer to [25].

(2) The CEC2013 contains 15 test questions:

- (i) F1–F3: separable functions
- (ii) F4–F11: partially separable functions
- (iii) F12–F14: overlapping functions ($D = 905$)
- (iv) F15: fully non-separable functions

For more detailed features, please refer to [26].

To make the experimental data more accurate, each experiment was run 25 times to record the statistical results. The solution error measure $((x) - (x^*))$ was recorded at the end of each run; x^* is the well-known global optimum of each function. The maximum number of iterations is set to $3.0E+6$ according to the default value of test suite.

5.1. Parametric Analysis. Most of the parameters of GI-ES are the same as reference [27]. In the algorithm, the number of population and the learning rate of gradient information are the parameters that need to be specified artificially.

In Section 4, we stated that GI-ES represents a mixed effect of three main components which are as follows: (1) the fitness scores were optimized for each sampling scheme for each GI-ES, (2) the gradient information is simulated using individual information, and the approximate gradient information is used to guide the search, and (3) GI-ES uses an approximate gradient for the search directions.

To further verify the performance of the algorithm, we analyzed each part of the algorithm to show the individual effect of each components. Table 1 illustrates the mean values of each component, and best values are marked in bold. It can be seen from the results that the GI-ES algorithm with three mechanisms works best.GI-ES (1): the fitness scores were optimized for each sampling scheme for each GI-ES. GI-ES (2): the gradient information is simulated using individual information, and the approximate gradient

Require: $f(x)$: objective function; μ_{init}: initial μ; $\Sigma_{\text{init}} = \mathbf{A}^{\top}\mathbf{A}$;
Ensure: optimal x^{*}
(1) initial $\sigma \longleftarrow \sqrt{[d]}|\det(\mathbf{A})|$ and $\mathbf{B} \longleftarrow \mathbf{A}/\sigma$;
(2) **repeat**
(3) **for** $k = 1 \ldots \lambda$ **do**
(4) draw sample $\mathbf{s}_k \sim \mathcal{N}(0, \mathrm{II})$;
(5) $\mathbf{z}_k \longleftarrow \mu + \sigma\mathbf{B}^{\top}\mathbf{s}_k$;
(6) evaluate the fitness value
(7) **end for**
(8) sort the sampling particles according to the fitness value and compute utilities function u_k
(9) compute approximate gradients according to (17)–(20)
(10) update parameters use the approximate gradient information
(11) update the approximate mean vector $\mu \longleftarrow \mu + \eta_{\delta} \cdot \sigma\mathbf{B} \cdot \nabla_{\delta} J$
(12) update the scalar step size $\sigma \longleftarrow \sigma \cdot \exp(\eta_{\sigma}/2 \cdot \nabla_{\sigma} J)$
(13) update the covariance factor $\mathbf{B} \longleftarrow \mathbf{B} \cdot \exp(\eta_{B}/2 \cdot \nabla_{\mathbf{B}} J)$
(14) **until** the termination criterion

Algorithm 1: GI-ES.

information is used to guide the search. GI-ES (3): GI-ES uses an approximate gradient for the search directions.

5.2. Evaluation Criteria. To evaluate the performance of GI-ES, we apply the same methods in [18] such that three evaluation criteria were used. The first is Formula One Score (FOS). Formula One Score was used in the latest LSGO competition (CEC2015). According to this criterion, the algorithms will be ranked from best to worst. Then, the top 10 ranks will get 25, 18, 15, 12, 10, 8, 6, 4, 2, and 1, respectively. Algorithms ranked outside the top 10 will get zero. Maximum values of R indicate better performance. The second and third are two non-parametric statistical hypothesis tests: Friedman test using $\alpha = 0.05$ as a significance level.

5.3. Performance Analysis Using CEC2010 and CEC2013. Statistical results of GI-ES using CEC2010 and CEC2013 are illustrated in Tables 2 and 3, respectively. Figure 1 illustrates the convergence behavior of GI-ES using sample functions from each class in CEC2013: f3 as fully separable, f8 and f11 as partially separable functions, f12 and f14 as overlapping functions where $D = 905$, and f15 as fully non-separable functions.

CMA-ES variants for solving large-scale optimization are used as comparison algorithms. Some other algorithms are not derived from CMA-ES but are the state-of-the-art ones, for example, CC-based differential evolution (DECC-G) [14], the multiple offspring sampling (MOS) [9]. The comparison algorithm is shown in Tables 4 and 5. All of these algorithms followed the same CEC2010 and CEC2013 guidelines. The results of the comparative experiment are recorded in Tables 6 and 7.

Tables 8 and 9 summarize the ranking for GI-ES and the state-of-the-art algorithms using Formula One Score (FOS). Tables 10 and 11 summarize the ranking obtained using Friedman's test.

Table 1: Mean values of GI-ES and three component names GI-ES (1), GI-ES (2), and GI-ES (3) on CEC2013.

Func.	GI-ES (1)	GI-ES (2)	GI-ES (3)	GI-ES
F1	$1.26E+05$	**6.62E + 03**	$7.34E+04$	$5.06E+04$
F2	$9.11E+02$	$2.92E+02$	$9.85E+03$	**2.10E − 24**
F3	$6.16E+00$	**8.09E − 13**	$9.36E+00$	$1.05E+01$
F4	$4.51E+09$	$5.07E+09$	$3.30E+11$	**1.39E + 08**
F5	$1.93E+06$	$1.25E+06$	$7.48E+06$	**1.09E + 06**
F6	$5.43E+03$	$5.47E+03$	**2.28E + 01**	$1.11E+06$
F7	$7.78E+01$	$3.59E+06$	$4.20E+09$	**6.34E + 01**
F8	$2.08E+12$	**2.04E + 10**	$6.21E+15$	$3.12E+11$
F9	$1.61E+08$	$8.60E+07$	$5.28E+08$	**1.35E + 08**
F10	$2.91E+04$	$8.46E+02$	**7.60E + 02**	$8.11E+07$
F11	$2.55E+08$	$2.21E+08$	$3.80E+11$	**5.14E + 05**
F12	$4.35E+03$	$2.39E+03$	$1.16E+11$	**5.13E + 01**
F13	$8.02E+08$	$9.04E+06$	$1.92E+08$	**9.87E + 04**
F14	$5.15E+08$	$6.88E+08$	$3.65E+07$	**4.58E + 06**
F15	$3.39E+06$	$2.86E+06$	$1.32E+06$	**5.56E + 05**

5.3.1. Formula One Score (FOS). As shown in Table 8 and 9, GI-ES ranked second in all comparison algorithms of CEC2010 using Formula One Score (FOS) and first for CEC2013. Regarding CEC2010, the best algorithm is MMO-CC with 237 points. Comparing the winners of CEC2010 and CEC2012: MA-SW-chains and MOS2012 get 115 and 168 points. This shows that GI-ES is competitive. Using CEC2013, GI-ES gets points, followed by MOS2013, and VGDE, with 218 and 196 points, respectively.

5.3.2. Friedman Test. According to Friedman test illustrated in Tables 10 and 11, GI-ES obtained the best ranking for both CEC2010 and CEC2013 benchmarks. Using CEC2010, GI-ES gets 5.33 points, jDEsps gets 6 points, MMO-CC gets 7.75, and MA-SW-chains gets 7.95 points. While using CEC2013, GI-ES gets 3 points, MOS2013 gets 3.57 points.

From the previous comparison, high ranking using Formula One Score (FOS) does not guarantee the same ranking using Friedman test.

Table 2: GI-ES statistical results using CEC2010, the statistical results for all functions include best, worst, median, mean, and standard deviation calculated over 25 runs.

Func	Best	Worst	Median	Mean	Std.
F1	$3.54E+03$	$3.88E+03$	$3.77E+03$	$3.71E+03$	$1.70E+02$
F2	$8.59E+02$	$1.18E+03$	$1.01E+03$	$1.02E+03$	$1.61E+02$
F3	$8.69E-01$	$1.33E+00$	$1.03E+00$	$1.10E+00$	$2.31E-01$
F4	$2.49E+09$	$2.73E+09$	$2.27E+09$	$2.61E+09$	$1.21E+08$
F5	$1.87E+07$	$4.41E+07$	$3.52E+07$	$3.14E+07$	$1.27E+07$
F6	$1.02E+01$	$1.02E+01$	$1.02E+01$	$1.02E+01$	$1.27E-02$
F7	$8.81E+04$	$1.32E+05$	$1.12E+05$	$1.10E+05$	$2.19E+04$
F8	$2.42E+05$	$3.30E+05$	$1.20E+05$	$2.86E+05$	$4.41E+04$
F9	$2.10E+03$	$4.54E+03$	$2.78E+03$	$3.32E+03$	$1.22E+03$
F10	$9.07E+02$	$1.13E+03$	$1.01E+03$	$1.02E+03$	$1.13E+02$
F11	$1.09E+01$	$1.34E+01$	$2.01E+01$	$1.21E+01$	$1.25E+00$
F12	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$
F13	$1.90E+00$	$2.23E+01$	$1.04E+01$	$1.21E+01$	$1.02E+01$
F14	$3.30E+03$	$6.36E+03$	$5.22E+03$	$4.83E+03$	$1.53E+03$
F15	$9.07E+02$	$1.15E+03$	$1.01E+03$	$1.03E+03$	$1.23E+02$
F16	$9.10E+00$	$3.37E+01$	$3.23E+01$	$2.14E+01$	$1.23E+01$
F17	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$
F18	$1.98E+02$	$2.20E+03$	$2.00E+03$	$1.20E+03$	$1.00E+03$
F19	$8.03E+04$	$8.99E+04$	$9.21E+04$	$8.51E+04$	$4.82E+03$
F20	$6.40E+01$	$1.01E+03$	$5.19E+02$	$4.39E+02$	$4.75E+02$

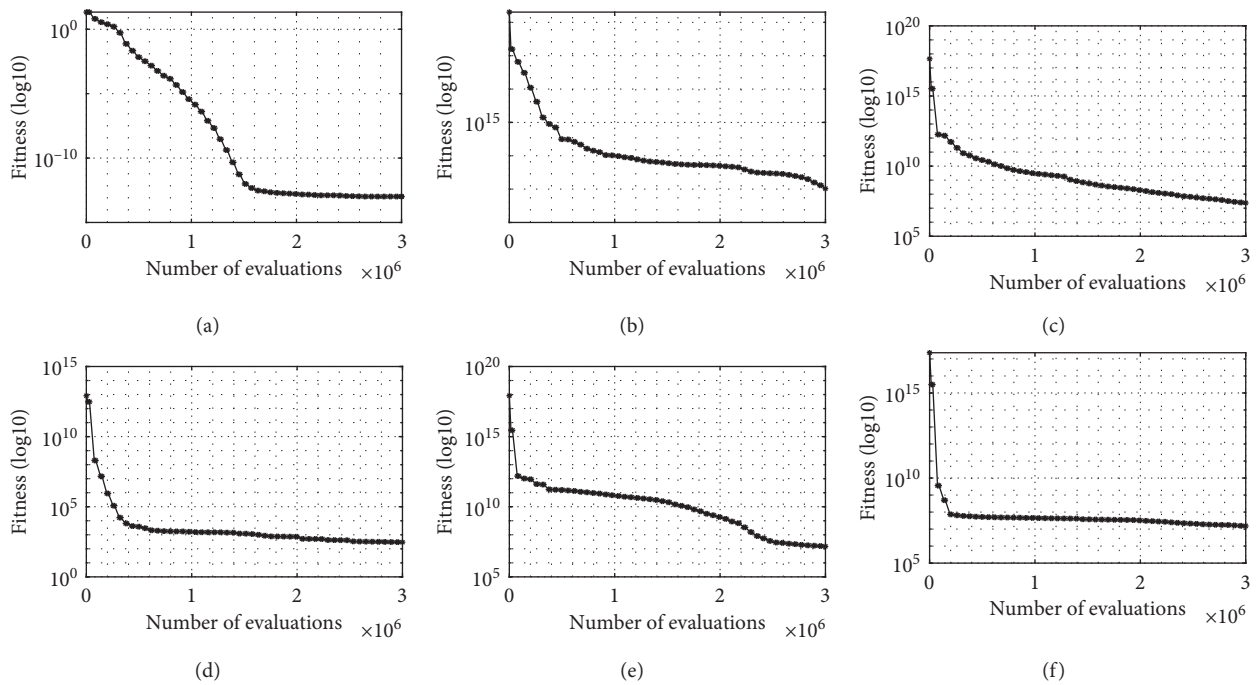

Figure 1: Convergence behavior of GI-ES using CEC2013. f3 as fully separable, f8 and f11 as partially separable functions, f12 and f14 as overlapping functions where $D = 905$, and f15 as fully non-separable functions. (a) Function # 3, (b) Function # 8, (c) Function # 11, (d) Function # 12, (e) Function # 14, and (f) Function # 15.

In the Friedman test, the critical value is 16.92, and p value is $1.64E-05$. These indicate that there are significant differences between different algorithms.

High ranking using Formula One Score (FOS) does not guarantee the same ranking using Friedman test. Because more weight will be given for the top positions.

5.4. CPU Computational Time. Previous experimental results have evaluated the effectiveness of GI-ES. This section illustrates the analysis of the algorithm. The runtime of the algorithm is shown in Table 12. Table 12 records the average running time of GI-ES and the baseline versions of CMA-ES and MA-SW-Chain CEC2010 LSGO. The dimensions are 1000.

Table 3: GI-ES statistical results using CEC2013, the statistical results for all functions include best, worst, median, mean, and standard deviation calculated over 25 runs.

Func	Best	Worst	Median	Mean	Std.
F1	$3.86E+04$	$6.26E+04$	$5.01E+04$	$5.06E+04$	$1.20E+04$
F2	$1.98E-24$	$2.22E-24$	$0.00E+00$	$2.10E-24$	$1.21E-25$
F3	$1.05E+01$	$1.05E+01$	$1.03E+01$	$1.05E+01$	$1.14E-02$
F4	$6.98E+07$	$2.08E+08$	$6.05E+12$	$1.39E+08$	$6.92E+07$
F5	$8.56E+05$	$1.32E+06$	$4.03E-25$	$1.09E+06$	$2.34E+05$
F6	$1.10E+06$	$1.12E+06$	$1.25E+03$	$1.11E+06$	$1.15E+04$
F7	$2.32E+01$	$1.04E+02$	$1.93E+01$	$6.34E+01$	$4.02E+01$
F8	$1.06E+11$	$5.18E+11$	$2.01E+11$	$3.12E+11$	$2.06E+11$
F9	$1.24E+08$	$1.46E+08$	$1.51E+08$	$1.35E+08$	$1.09E+07$
F10	$8.06E+07$	$8.16E+07$	$8.10E+07$	$8.11E+07$	$5.30E+05$
F11	$3.02E+05$	$7.26E+05$	$3.23E+05$	$5.14E+05$	$2.12E+05$
F12	$4.12E+01$	$6.14E+01$	$2.31E+00$	$5.13E+01$	$1.01E+01$
F13	$9.15E+04$	$1.06E+05$	$6.21E+04$	$9.87E+04$	$7.18E+03$
F14	$4.24E+06$	$4.92E+06$	$4.35E+06$	$4.58E+06$	$3.39E+05$
F15	$3.22E+05$	$7.90E+05$	$5.01E+05$	$5.56E+05$	$2.34E+05$

Table 4: Compared algorithms using CEC2010.

#	Algorithm	Published year
1	SDENS [28]	2010
2	jDElsgo [10]	2010
3	DECC-DML [12]	2010
4	MA-SW-chains [29]	2010
5	DASA [8]	2010
6	EOEA [7]	2010
7	LMDEa [30]	2012
8	jDEsps [4]	2012
9	MOS2012 [9]	2012
10	CCGS [11]	2012
11	DM-HDMR [31]	2014
12	HACC-D [32]	2014
13	DISCC [33]	2016
14	EADE [34]	2017
15	ANDE [35]	2017
16	SEE [36]	2018
17	MMO-CC [37]	2018

Table 5: Compared algorithms using CEC2013.

#	Algorithm	Published year
1	MOS2013 [13]	2013
2	SACC [15]	2013
3	DECC-CG [14]	2013
4	VGDE [38]	2014
5	IHDELS [39]	2015
6	CBCC3-DG2 [40]	2016
7	CRO [41]	2016
8	CCFR-I [42]	2017
9	CCFRI-DG2 [42]	2017

It can be seen from the results that, for the separable functions f1–f3, the runtime of GI-ES is slightly better than that of MA-SW-Chain and CMA-ES. The run-time difference between the three algorithms is not obvious. For partially separable functions (f4–f18), the running time of GI-ES is significantly better than that of CMA-ES and MA-SW-Chains. In terms of fully non-separable functions, CMA-ES performs better than GI-ES. In general, GI-ES offers better performance on the part of the separability problem.

Table 6: Experimental comparisons between GI-ES and state-of-the-art algorithms using CEC2010.

Func.	GI-ES	EADE	ANDE	LMDEa	SDENS	jDElsgo	DECC-DML	MA-SW	DISCC
F1	3.71E+03	4.70E−22	3.72E−26	1.35E−23	5.73E−06	8.86E−20	1.93E−25	2.10E−14	9.77E−25
F2	1.02E+03	4.16E+02	3.72E+02	6.97E+02	2.21E+03	1.25E−01	2.17E+02	8.10E+02	5.07E+02
F3	1.10E+00	6.25E−14	7.46E−14	6.44E−01	2.70E−05	3.81E−12	1.18E−13	7.28E−13	7.77E−14
F4	2.61E+09	1.08E+11	5.13E+11	2.08E+11	5.11E+12	8.06E+10	3.58E+12	3.53E+11	1.26E+12
F5	3.14E+07	8.79E+07	9.19E+07	6.62E+07	1.18E+08	9.72E+07	2.99E+08	1.68E+08	2.35E+08
F6	1.02E+01	1.90E+01	1.64E+00	2.63E−01	2.02E−04	1.70E−08	7.93E+05	8.14E+04	2.13E+06
F7	1.10E+05	2.11E−01	1.80E+00	2.45E−01	1.20E+08	1.31E−02	1.39E+08	1.03E+02	6.45E+05
F8	2.86E+05	**2.26E−04**	1.25E+07	3.61E−04	5.12E+07	3.15E+06	3.46E+07	1.41E+07	7.54E+07
F9	3.32E+03	3.67E+07	4.12E+07	2.64E+07	5.63E+08	3.11E+07	5.92E+07	1.41E+07	6.08E+07
F10	1.02E+03	2.62E+03	3.06E+03	2.80E+03	6.87E+03	2.64E+03	1.25E+04	2.07E+03	2.27E+03
F11	1.21E+01	1.14E+02	8.95E+01	1.19E+01	2.21E+02	2.20E+01	1.80E−13	3.80E+01	8.20E−01
F12	0.00E+00	2.80E+04	4.89E+04	1.83E+04	4.13E+05	1.21E+04	3.80E+06	3.62E−06	3.34E+04
F13	**1.21E+01**	1.01E+03	1.11E+03	5.95E+02	2.19E+03	7.11E+02	1.14E+03	1.25E+03	1.31E+03
F14	4.83E+03	1.46E+08	1.87E+08	8.63E+07	1.88E+09	1.69E+08	1.89E+08	3.11E+07	2.08E+08
F15	1.03E+03	3.18E+03	3.19E+03	5.63E+03	7.32E+03	5.84E+03	1.54E+04	2.74E+03	5.54E+03
F16	2.14E+01	3.00E+02	3.04E+02	3.87E+02	4.08E+02	1.44E+02	5.08E−02	9.98E+01	1.98E+01
F17	**0.00E+00**	1.52E+05	2.02E+05	2.14E+05	1.08E+06	1.02E+05	6.54E+06	1.24E+00	1.81E+05
F18	**1.20E+03**	2.26E+03	2.35E+03	1.68E+03	3.08E+04	1.85E+03	2.47E+03	1.30E+03	5.16E+03
F19	8.51E+04	1.29E+06	1.63E+06	4.42E+05	8.80E+05	2.74E+05	1.59E+07	2.85E+05	1.71E+06
F20	**4.39E+02**	2.10E+03	2.20E+03	1.38E+03	9.90E+02	1.53E+03	9.91E+02	1.07E+03	1.94E+03
Func.	DASA	EOEA	jDEsps	MOS2012	DM-HDMR	HACC-D	CCGS	SEE	MMO-CC
F1	1.52E−21	2.20E−23	4.10E−23	**0.00E+00**	2.34E+01	1.99E−27	1.83E−22	6.99E−11	**0.00E+00**
F2	8.48E+00	3.62E−01	1.10E+02	1.97E+02	4.36E+03	**1.43E−14**	4.44E−02	8.77E+03	1.43E+03
F3	7.20E−11	1.67E−13	1.30E−13	1.12E+00	1.67E+01	3.45E−14	1.91E−01	1.99E+01	**0.00E+00**
F4	5.05E+11	2.86E+12	8.15E+11	1.91E+10	6.96E+11	1.55E+12	1.79E+12	2.58E+11	**7.64E+06**
F5	6.20E+08	2.24E+07	7.71E+07	6.81E+08	1.45E+08	1.96E+08	**1.97E+07**	5.85E+08	3.34E+08
F6	1.97E+07	3.85E+06	5.58E−03	1.99E+07	1.63E+01	**3.55E−09**	2.88E+06	1.99E+07	5.77E−01
F7	7.78E+00	1.24E+02	5.77E+05	0.00E+00	2.91E+05	**3.87E−07**	1.37E+02	3.14E−02	2.41E+10
F8	4.98E+07	1.01E+07	1.52E+06	1.12E+06	4.41E+07	7.44E+07	2.81E+07	1.82E+06	2.63E+08
F9	3.60E+07	4.63E+07	2.31E+04	5.75E+06	5.20E+07	3.32E+07	5.53E+07	2.67E+07	**8.99E+01**
F10	7.29E+03	**1.00E+03**	1.85E+03	7.86E+03	4.49E+03	1.30E+04	4.74E+03	1.27E+04	1.63E+03
F11	1.98E+02	3.18E+01	1.94E−05	1.99E+02	1.10E+01	**7.82E−14**	2.99E+01	2.19E+02	2.99E+00
F12	1.78E+03	2.61E+04	1.57E+04	**0.00E+00**	1.97E+03	1.31E+06	5.35E+03	2.60E+02	0.00E+00
F13	1.21E+03	1.24E+03	1.86E+02	1.36E+03	3.35E+06	1.96E+03	1.51E+03	7.12E+02	3.05E+04
F14	1.00E+08	1.65E+08	3.85E+05	1.52E+07	3.41E+08	9.21E+07	1.35E+08	9.88E+07	**0.00E+00**
F15	1.45E+04	2.14E+03	5.50E+03	1.54E+04	5.95E+03	1.56E+04	**1.74E+03**	1.50E+04	2.05E+03
F16	3.97E+02	8.26E+01	4.97E+00	3.97E+02	1.24E−06	**1.95E−11**	3.11E+01	3.97E+02	8.87E+00
F17	1.03E+04	7.93E+04	5.52E+04	4.66E−05	4.03E+04	1.42E+06	1.48E+04	7.40E+03	**0.00E+00**
F18	4.92E+03	2.94E+03	**9.73E+02**	3.91E+03	8.40E+03	4.02E+03	3.13E+03	3.14E+03	3.37E+04
F19	8.34E+05	1.84E+06	8.00E+05	**3.41E+04**	1.71E+06	1.87E+07	5.93E+05	7.13E+05	1.54E+07
F20	1.13E+03	1.97E+03	8.79E+02	5.31E+02	2.45E+06	1.51E+03	1.31E+03	1.43E+03	1.10E+03

FE = 3.0E+06 bold font represents the best result.

Table 7: Experimental comparisons between GI-ES and state-of-the-art algorithms using CEC2013.

Func.	GI-ES	MOS2013	DECC-CG	CBCC3-DG2	CCFR-IDG2	CCFR-I	CRO	IHDELS	VGDE	SACC
F1	5.06E+04	0.00E+00	2.03E−13	8.65E+05	2.00E−05	1.30E−05	1.84E+06	4.34E−28	0.00E+00	2.73E−24
F2	**2.10E−24**	8.32E+02	1.03E+03	1.41E+04	3.60E+02	5.50E−01	9.84E+02	1.32E+03	4.56E+01	7.06E+02
F3	1.05E+01	**9.17E−13**	2.87E−10	2.06E+01	2.10E+01	2.00E+01	2.01E+01	2.01E+01	3.98E−13	1.11E+00
F4	1.39E+08	1.74E+08	2.60E+10	3.39E+07	9.60E+07	**4.50E+07**	1.55E+10	3.04E+08	5.96E+08	4.56E+10
F5	1.09E+06	6.94E+06	7.28E+14	2.14E+06	2.80E+06	**2.50E+06**	2.38E+07	9.59E+06	3.00E+06	7.74E+06
F6	1.11E+06	1.48E+05	**4.85E+04**	1.05E+06	1.10E+06	1.10E+06	1.06E+06	1.03E+06	1.31E+05	2.47E+05
F7	**6.34E+01**	1.62E+04	6.07E+08	2.95E+07	2.00E+07	8.60E+06	2.78E+08	3.46E+04	1.85E+03	8.98E+07
F8	3.12E+11	8.00E+12	4.26E+14	**6.74E+10**	7.00E+10	9.60E+09	4.56E+14	1.36E+12	7.00E+14	1.20E+15
F9	**1.35E+08**	3.83E+08	4.27E+08	1.70E+08	1.90E+08	1.90E+08	5.27E+08	6.74E+08	2.31E+08	5.98E+08
F10	8.11E+07	**9.02E+05**	1.10E+07	9.28E+07	9.50E+07	9.50E+07	9.44E+07	9.16E+07	1.57E+02	2.95E+07
F11	**5.14E+05**	5.22E+07	2.46E+11	7.70E+08	4.00E+08	3.30E+08	2.91E+10	1.07E+07	7.52E+07	2.78E+09
F12	**5.13E+01**	2.47E+02	1.04E+03	5.81E+07	1.60E+09	6.00E+08	3.69E+03	3.77E+02	2.52E+03	8.73E+02
F13	**9.87E+04**	3.40E+06	3.42E+10	6.03E+08	1.20E+09	9.30E+08	5.33E+09	3.80E+06	1.36E+09	1.78E+09
F14	**4.58E+06**	2.56E+07	6.08E+11	1.11E+09	3.40E+09	2.10E+09	6.08E+10	1.58E+07	2.29E+10	1.75E+10
F15	**5.56E+05**	2.35E+06	6.05E+07	7.11E+06	9.80E+06	8.20E+06	1.88E+07	2.81E+06	3.44E+06	2.01E+06

FE = 3.0E+06, bold font represents the best result.

Table 8: Ranks of 18 algorithms using CEC2010 according to FOS (the top three are in bold).

GI-ES	EADE	ANDE	LMDEa	SDENS	jDElsgo	DECC-DML	MA-SW-Chains	DISCC
226	106	76	125	29	125	71	115	50
DASA	EOEA	jDEsps	MOS2012	DM-HDMR	HACC-D	CCGS	SEE	MMO-CC
49	94	195	**168**	43	164	98	79	**237**

Table 9: Ranks of 10 algorithms using CEC2013 according to FOS (the top three are in bold).

GI-ES	MOS2013	DECC-CG	CBCC3-DG2	CCFR-IDG2	CCFR-I	CRO	IHDELS	VGDE	SACC
278	**218**	103	150	120	154	55	157	**196**	111

Table 10: Ranking of GI-ES and state-of-the-art algorithms according to Friedman test using CEC2010.

GI-ES	jDEsps	jDElsgo	MMO-CC	LMDEa	MA-SW-Chains	MOS2012	EADE	CCGS
5.33	6	7.75	7.75	7.8	7.95	8.63	8.8	9.4
EOEA	HACC-D	ANDE	DASA	SEE	DISCC	DECC-DML	DM-HDMR	SDENS
9.6	9.85	9.9	11.1	11.28	11.63	11.68	12.48	14.1

Table 11: Ranking of GI-ES and state-of-the-art algorithms according to Friedman test using CEC2013.

GI-ES	MOS2013	DECC-CG	CBCC3-DG2	CCFR-IDG2	CCFR-I	CRO	IHDELS	VGDE	SACC
3	3.57	7.2	5.6	6.3	5.17	8.3	5.17	4.43	6.27

Table 12: CPU computational time of improved CMA-ES and non-improved CMA-ES.

$D = 1000$	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
MA-SW-Chain	166.87	218.9	197.29	199.92	260.2	229.17	169.57	165.96	431.78	478.9
CMA-ES	159.14	216.73	184.41	141.91	232.9	237.31	236.01	136.01	186.61	198.43
GI-ES	147.96	213.04	180.72	98.22	189.21	193.62	192.32	92.32	142.92	154.74
$D = 1000$	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20
MA-SW-Chain	472.41	169.31	169.96	687.67	735.8	725.51	174.55	180.11	163.41	174.07
CMA-ES	308.27	106.25	137.12	233.54	372.58	277.15	36.5	38.39	136.57	138.22
GI-ES	251.96	92.54	93.41	189.85	328.89	333.46	92.81	94.7	192.88	194.53

MA-SW-Chain is also used as a comparison. The table records the average time of 51 independent runs of the three algorithms when the dimension is 1000. The maximum number of iterations is $10E06$.

6. Conclusion

This research has shown that the use of guiding gradient information improves the performance of evolutionary computations. The problem of low utilization of historical information by ES was solved by guiding the information generated by the distribution of next-generation solutions. The guidance information was obtained by approximating the gradient. This strategy not only increased the diversity of information, but also made full use of the optimal information in the heuristic algorithm. The theoretical analysis and experimental results showed that this method incorporating guidance information is accurate and stable.

The experimental results showed that the use of guiding information is effective. The algorithm was also compared with other state-of-the-art meta-heuristic algorithms and demonstrated good average performance and pair-wise comparison performance across a wide range of test functions.

The experiments showed that this algorithm is an effective global optimization method for large-scale problems, which makes it applicable to a large number of practical applications. The principle behind the use of guidance information is simple, but effective, and has a certain guiding significance for heuristic optimization algorithms.

Data Availability

The data are available at https://titan.csit.rmit.edu.au/~e46507/publications/lsgo-cec10.pdf.

Conflicts of Interest

The author declares that there are no conflicts of interest.

References

[1] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[2] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111–128, 2011.

[3] L.-Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, Hong Kong, China, June 2008.

[4] S.-Z. Zhao, P. N. Suganthan, and S. Das, "Self-adaptive differential evolution with multi-trajectory search for large-scale optimization," *Soft Computing*, vol. 15, no. 11, pp. 2175–2185, 2011.

[5] 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.

[6] D. Molina, M. Lozano, and F. Herrera, "Ma-sw-chains: memetic algorithm based on local search chains for large scale continuous global optimization," in *Proceedings of the IEEE Congress on evolutionary computation*, pp. 1–8, Barcelona, Spain, July 2010.

[7] Y. Wang and B. Li, "Two-stage based ensemble optimization for large-scale global optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, Barcelona, Spain, July 2010.

[8] P. Korošec, J. Šilc, and B. Filipič, "The differential ant-stigmergy algorithm," *Information Sciences*, vol. 192, pp. 82–97, 2012.

[9] A. LaTorre, S. Muelas, and J.-M. Pena, "Multiple offspring sampling in large scale global optimization," in *Proceedings of the 2012 IEEE Congress on Evolutionary Computation*, Brisbane, Australia, June 2012.

[10] J. Brest, A. Zamuda, I. Fister, and M. S. Maučec, "Large scale global optimization using self-adaptive differential evolution algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, Barcelona, Spain, July 2010.

[11] K. Zhang and B. Li, "Cooperative coevolution with global search for large scale global optimization," in *Proceedings of the 2012 IEEE Congress on Evolutionary Computation*, pp. 1–7, Brisbane, Australia, June 2012.

[12] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, Barcelona, Spain, July 2010.

[13] A. LaTorre, S. Muelas, and J.-M. Peña, "Large scale global optimization: experimental results with mos-based hybrid algorithms," in *Proceedings of the 2013 IEEE congress on evolutionary computation*, pp. 2742–2749, Cancun, Mexico, June 2013.

[14] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[15] F. Wei, Y. Wang, and Y. Huo, "Smoothing and auxiliary functions based cooperative coevolution for global optimization," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pp. 2736–2741, Cancun, Mexico, June 2013.

[16] H. Ge, M. Zhao, Y. Hou et al., "Bi-space interactive cooperative coevolutionary algorithm for large scale black-box optimization," *Applied Soft Computing*, vol. 97, Article ID 106798, 2020.

[17] D. Molina, A. LaTorre, and F. Herrera, "Shade with iterative local search for large-scale global optimization," in *Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Rio de Janeiro, Brazil, July 2018.

[18] A. A. Hadi, A. W. Mohamed, and K. M. Jambi, "Lshade-spa memetic framework for solving large-scale optimization problems," *Complex & Intelligent Systems*, vol. 5, no. 1, pp. 25–40, 2019.

[19] V. A. Shim, K. C. Tan, and K. K. Tan, "A hybrid adaptive evolutionary algorithm in the domination-based and decomposition-based frameworks of multi-objective optimization," in *Proceedings of the 2012 IEEE Congress on Evolutionary Computation*, June 2012.

[20] A. LaTorre, S. Muelas, and J.-M. Peña, "A comprehensive comparison of large scale global optimizers," *Information Sciences*, vol. 316, pp. 517–549, 2015.

[21] N. Hansen, "The CMA evolution strategy: a tutorial," 2016, http://arxiv.org/abs/1604.00772.

[22] A. Bordes, L. Bottou, and P. Gallinari, "Sgd-qn: careful quasi-Newton stochastic gradient descent," *Journal of Machine Learning Research*, vol. 10, pp. 1737–1754, 2009.

[23] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, http://arxiv.org/abs/1412.6980.

[24] J. Grefenstette, "Rank-based selection," *Evolutionary Computation*, vol. 1, pp. 187–194, 2000.

[25] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization," Technical Report, Nature Inspired Computation and Applications Laboratory, Hefei, China, 2009.

[26] X. Li, K. Tang, M. N. Omidvar, Z. Yang, K. Qin, and H. China, "Benchmark functions for the CEC 2013 special session and competition on large-scale global optimization," *Gene*, vol. 7, no. 33, p. 8, 2013.

[27] J. Jin, C. Yang, and Y. Zhang, "An improved CMA-ES for solving large scale optimization problem," in *Advances in Swarm Intelligence*, Y. Tan, Y. Shi, and M. Tuba, Eds., , pp. 386–396, Springer International Publishing, 2020.

[28] H. Wang, Z. Wu, S. Rahnamayan, and D. Jiang, "Sequential de enhanced by neighborhood search for large scale global optimization," in *Proceedings of the IEEE congress on evolutionary computation*, pp. 1–7, Barcelona, Spain, July 2010.

[29] D. Molina, M. Lozano, and F. Herrera, "MA-SW-chains: memetic algorithm based on local search chains for large scale continuous global optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, Barcelona, Spain, July 2010.

[30] T. Takahama and S. Sakai, "Large scale optimization by differential evolution with landscape modality detection and a diversity archive," in *Proceedings of the 2012 IEEE Congress on Evolutionary Computation*, pp. 1–8, Brisbane, Australia, June 2012.

[31] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Cooperative co-evolution with a new decomposition method for large-scale optimization," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1285–1292, Beijing, China, July 2014.

[32] S. Ye, G. Dai, L. Peng, and M. Wang, "A hybrid adaptive coevolutionary differential evolution algorithm for large-scale optimization," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1277–1284, Beijing, China, July 2014.

[33] G. Dai, X. Chen, L. Chen, M. Wang, and L. Peng, "Cooperative coevolution with dependency identification grouping for large scale global optimization," in *Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 5201–5208, Vancouver, Canada, July 2016.

[34] A. W. Mohamed, "Solving large-scale global optimization problems using enhanced adaptive differential evolution

algorithm," *Complex & Intelligent Systems*, vol. 3, no. 4, pp. 205–231, 2017.

[35] A. W. Mohamed and A. S. Almazyad, "Differential evolution with novel mutation and adaptive crossover strategies for solving large scale global optimization problems," *Applied Computational Intelligence and Soft Computing*, vol. 2017, Article ID 7974218, 18 pages, 2017.

[36] P. Yang, K. Tang, and X. Yao, "Turning high-dimensional optimization into computationally expensive optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 143–156, 2017.

[37] X. Peng, Y. Jin, and H. Wang, "Multimodal optimization enhanced cooperative coevolution for large-scale optimization," *IEEE Transactions on Cybernetics*, vol. 49, no. 9, pp. 3507–3520, 2018.

[38] F. Wei, Y. Wang, and T. Zong, "Variable grouping based differential evolution using an auxiliary function for large scale global optimization," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1293–1298, IEEE, Beijing, China, July 2014.

[39] D. Molina and F. Herrera, "Iterative hybridization of de with local search for the cec'2015 special session on large scale global optimization," in *Proceedings of the 2015 IEEE congress on evolutionary computation (CEC)*, pp. 1974–1978, IEEE, Sendai, Japan, May 2015.

[40] M. N. Omidvar, B. Kazimipour, X. Li, and X. Yao, "CBCC3—a contribution-based cooperative co-evolutionary algorithm with improved exploration/exploitation balance," in *Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 3541–3548, Vancouver, Canada, July 2016.

[41] S. Salcedo-Sanz, C. Camacho-Gómez, D. Molina, and F. Herrera, "A coral reefs optimization algorithm with substrate layers and local search for large scale global optimization," in *Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 3574–3581, Vancouver, Canada, July 2016.

[42] M. Yang, M. N. Omidvar, C. Li et al., "Efficient resource allocation in cooperative co-evolution for large-scale global optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 493–505, 2016.