Low-dimensional Space Modeling-based Differential Evolution for Large Scale Global Optimization Problems

Thiago Henrique Lemos Fonseca[1], Silvia Modesto Nassar[1], Alexandre César Muniz de Oliveira[2] and Bruno Agard[3]

***Abstract*—Large-Scale Global Optimization (LSGO) has been an active research field. Part of this interest is supported by its application to cutting-edge research such as Deep Learning, Big Data, and complex real-world problems such as image encryption, real-time traffic management, and more. However, the high dimensionality makes solving LSGO a significant challenge. Some recent research deal with the high dimensionality by mapping the optimization process to a reduced alternative space. Nonetheless, these works suffer from the changes in the search space topology and the loss of information caused by the dimensionality reduction. This paper proposes a hybrid metaheuristic, so-called LSMDE (Low-dimensional Space Modeling-based Differential Evolution), that uses the Singular Value Decomposition to build a low-dimensional search space from the features of candidate solutions generated by a new SHADE-based algorithm (GM-SHADE). GM-SHADE combines a Gaussian Mixture Model (GMM) and two specialized local algorithms: MTS-LS1 and L-BFGS-B, to promote a better exploration of the reduced search space. GMM mitigates the loss of information in mapping high-dimensional individuals to low-dimensional individuals. Furthermore, the proposal does not require prior knowledge of the search space topology, which makes it more flexible and adaptable to different LSGO problems. The results indicate that LSMDE is the most efficient method to deal with partially separable functions compared to other state-of-the-art algorithms and has the best overall performance in two of the three proposed experiments. Experimental results also show that the new approach achieves competitive results for non-separable and overlapping functions on the most recent test suite for LSGO problems.**

***Index Terms*—Gaussian Mixture Model, Dimensionality Reduction, Differential Evolution, Singular Value Decomposition**

I. Introduction

Large Scale Global Optimization (LSGO) deals with optimization problems with a high number of decision variables [1]. Its study is of great relevance to Computer Science, not only for developing new evolutionary algorithms for these problems but also for encouraging lines of research closely linked to large-scale computational projects, such as Distributed Systems and Parallel Algorithms [2], [3]. The applicability of LSGO to subjects of intense studies, such as *Deep Learning*, *Big Data*, among others, is also a factor that attracts interest in the design of increasingly efficient algorithms [4]–[6]. However, several factors make solving LSGO problems a major challenge. For example, the evaluation of a large-scale problem is often computationally expensive [7]. This high cost is often the case in many real-world problems, such as image encryption [8], [9], real-time traffic management [10], construction engineering [6], and others.

Another factor contributing to the complexity of large-scale optimization problems is the interaction among variables. This interaction prevents them from being independently optimized to find the global optimum of an objective function [2], [11]. In the Continuous Optimization literature, the interaction between variables is commonly referred to as Non-separability [12], [13] and in Evolutionary Computing literature; this phenomenon is named *Epistasis* [14]. Decomposition algorithms based on Cooperative Coevolution (CC) are often employed to split the decision variables of the problem into several subproblems, each containing a subset of the original dimensions [2], [15], [16]. However, partitioning the dimensions could reduce the search space coverage since the optimization result combines multiple optimized subspaces. In addition, identifying the best grouping of dimensions requires higher computational cost evaluations [11], [15], [17].

Non-Decomposition algorithms have been developed over the past few years to mitigate the variable dependency problem [2], [18], [19]. These approaches improve the exploration of high-dimensional search spaces by creating new evolutionary operators, initialization methods, specialized local searches, and strategies that optimize the multidimensional space. This approach is responsible for developing one of the best current algorithms for LSGO problems [20], [21]. Nonetheless, Non-Decomposition algorithms still need to deal directly with the problems caused by the exponential increase in the search space, which degrades the benefits achieved by the developed specialized strategies over time [15], [22]. Recently, [23] and [24] demonstrated that it is possible to reduce the effects of dimensionality through optimization in a dimensionally reduced search space. The results, although encouraging, relied on a preliminary analysis of the search space topology to develop dimensionality reduction algorithms specific to each type of space, and it was not possible to extend them to general LSGO problems [18].

This paper proposes a new non-decomposition approach based on the so-called Success-History Based Parameter

[1]Thiago Henrique Lemos Fonseca and Silvia Modesto Nassar are with Universidade Federal de Santa Catarina (UFSC) - Florianópolis, SC, Brazil. thiago.lemos@ufsc.br, silvia.nassar@ufsc.br

[2]Alexandre César Muniz de Oliveira is with Universidade Federal do Maranhão (UFMA), São Luís, MA, Brazil and also with School of Computer Science and Information Technology, University College Cork, CO, Ireland. alexandre.cesar@ufma.br, amunizdeoliveira@ucc.ie

[3]Bruno Agard is with Laboratoire en Intelligence des Données, Polytechnique Montréal, Montréal (Québec), Canada. bruno.agard@polymtl.ca

Adaptation for Differential Evolution (SHADE) algorithm. Our algorithm uses a dimensionality reduction method named Singular Value Decomposition (SVD) that builds the low-dimensional search space from the features of the current population of candidate solutions generated by the SHADE-based algorithm. Therefore, our proposal does not require prior knowledge about the search space to explore it. Gaussian Mixture Model (GMM) is used with SHADE to promote the search for optimal solutions in a low-dimensional space. GMM mitigates the error in mapping high-dimensional individuals to low-dimensional individuals by soft clustering using probability distributions. The candidate's optimal solution can then be reconverted back to the original high-dimensional space where a local search method can explore its neighboring regions.

For experimentation, we considered the most recent LSGO test suite conducted at the 2013 IEEE Congress on Evolutionary Computation (CEC'2013) [25] and new evaluation criteria held at the 2019 IEEE Congress on Evolutionary Computation (CEC'2019) [26].

CEC'2013 benchmark is a recognized benchmark for comparing the selected algorithms on a wide variety of functions with different design perspectives, including nonuniform subcomponent sizes, imbalances in subcomponent contributions, overlapping subcomponents, ill-conditioning, symmetry breaking, and irregularities [27]. Our proposed algorithm, called Low-dimensional Space Modeling-based Differential Evolution (LSMDE), is evaluated against seven state-of-the-art algorithms that have also achieved competitive results in competitions on the CEC'2013 large-scale global optimization (LSGO) test suite over the past few years. The experimental results show that LSMDE has better performance on the partially separable functions than other state-of-the-art LSGO algorithms. The results also showed promising performance of the LSMDE on non-separable and overlapping functions.

In the paper, we provide the following main contributions: (i) we quantify the gain in terms of average fitness values that can be achieved with a LSMDE optimizer using the most recent test suite for LSGO problems; (ii) we provide valuable indications that makes the proposed approach suitable for partially separable optimization problems ; (iii) we propose an innovative way of applying Dimension Reduction to Large-Scale Optimization problems opening up possibilities for applications of different algorithms from the same scope (PCA, t-SNE).

In the following, Section II presents the related works. Section III describes in detail the proposed algorithm. Section IV presents the experimental design indicated by the *IEEE Task Force on Large-Scale Global Optimization* and our results. Section V presents the statistical analysis that supports our results. Finally, Section VI contains conclusions and further research.

II. Related Works

Several specific algorithms have been proposed to deal with Large Scale Global Optimization (LSGO) problems in the last decade. This problem has two main approaches: decomposition algorithms based on Cooperative Coevolution (CC) and non-decomposition algorithms. Decomposition-based algorithms use a divide-and-conquer approach by decomposing LSGO problems into several low-dimensional subcomponents. In contrast, non-decomposition algorithms improve the exploration of d-dimensional search spaces by creating new evolutionary operators and specialized local search (LS) methods [28]

The cooperative coevolution (CC) method, proposed by [29], decomposes the original large-scale problem into a certain number of one-dimensional subproblems before its evolution. A subcomponent optimizer can evolve a subpopulation for each subproblem in a round-robin fashion. A series of variants have been developed under the traditional CC in conjunction with different evolutionary strategies. Such as CCGS [30], CC-CMA-ES [31], SACC [2], CCFR [32], and many others.

Despite the improvement in computational performance provided by the constant decomposition algorithms based on CC, such gains generally do not offset the cost of evaluations required for variable dependency detection. According to [33], this approach is not considered competitive enough compared to algorithms specially designed for LSGO. Investing in mitigating the limitations of decomposition-based algorithms led to the investigation of other strategies. An interesting work developed by [34] describes the whole process of creating a competitive hybrid algorithm specific for LSGO with the experimental design to the final statistical validation of the results. These results show that a good experimental design can find a combination of algorithms that outperforms any previous decomposition-based algorithms, automatically selecting the most suitable heuristic for each function and search phase.

Among these hybrid approaches, algorithms based on Swarm Intelligence (SI) and Differential Evolution (DE) have gained prominence for LSGO problems due to their simplicity and efficiency [28]. One example of this combination is C-DEEPSO, proposed by [35]. C-DEEPSO is a hybrid metaheuristic that incorporates distinctive features of Particle Swarm Optimization (PSO), Differential Evolution (DE), and Evolutionary Programming (EP). The distinctive feature of C-DEEPSO consists of improved assimilation of the optimization landscape. To take advantage of the information collected by the population throughout the search, C-DEEPSO relies on a collective memory instead of multiple and independent memories that encompass the search experience of each individual. The results indicate that the C-DEEPSO is an efficient and competitive algorithm for tackling LSGO problems.

Differential Evolution (DE) has proven to be a promising approach in decomposition-based and non-decomposition-based algorithms, with several significant improvements made over the last few years [36]. Researches have proposed new mutation strategies to improve the optimization performance of DE algorithms [37]. For example, [38] proposed an enhanced adaptive differential evolution (EADE) algorithm that utilizes the information of good and bad vectors in the DE population through a new mutation rule. Already, [39] uses a new mutation rule to balance the global exploration ability and the local exploitation tendency and enhance the convergence rate of an algorithm so-called ANDE. The

comparison results between EADE and ANDE and the other state-of-art algorithms indicate that both algorithms are highly competitive for LSGO problems. [40] also used a new mutation strategy based on neighborhood, quantic computing characteristics, and a CC method to propose an improved DE with higher convergence accuracy and stability for high-dimensional problems. Another interesting strategy is to combine DE algorithms with LS methods. [41] proposed combining the DE's exploratory component with the LS method's exploitative factor in an algorithm named IHDELS. [42] developed the SHADE, a DE that uses a success history-based parameter adaptation technique. In the SHADE, a memory pool stores the successful parameter values of scaling factor F and crossover rate CR. For every individual, F and CR values are calculated by selecting a random location from historical memory and using the corresponding values with Cauchy and normal distribution, respectively. This algorithm became a reference for other algorithms that came later due to its promising results in LSGO benchmark suites.

Few DE algorithms have succeeded in LSGO problems as the SHADE-ILS proposed by [21]. Winner of the *IEEE CEC'2018 Special Session and Competition on Large-Scale Global Optimization*, SHADE-ILS is a SHADE-based hybrid algorithm that combines a modern Differential Evolution algorithm with a local search method chosen from a set of different search methods. The selection of the local search method is dynamic. SHADE-ILS considers the improvement obtained by each search in the previous intensification phase to identify the most suitable method for each iteration. [43] proposed a SHADE-based algorithm with linear population size reduction and semi-parameter adaptation (MLSHADE-SPA) where dimensions are randomly divided into groups and solved separately. Subsequently, an adaptive local search method was developed for MLSHADE-SPA, achieving promising results on non-separable functions and overlapping functions [17]. Another efficient SHADE-based algorithm was the GL-SHADE proposed by [20]. Two populations are used in GL-SHADE, each evolving differently, allowing them to complement each other during the search process. The first population is responsible for exploring the search space, while the second is responsible for exploiting it. GL-SHADE outperformed the SHADE-ILS in many IEEE LSGO test problems [20].

The hybridization of metaheuristics and local search has a fundamental and compelling place in LSGO research [6]. This influence is more evident when considering related papers and competitions in special sessions such as the CEC Special Sessions and Competitions organized by The IEEE Congress on Evolutionary Computation (IEEE CEC).

The first winner of the LSGO competitions was Multiple Trajectory Search (MTS), a combination of three local searches presented in CEC'2008 [44]. MTS uses three methods to find candidate solutions neighboring the current solution [6]. Among these local search methods, MTS-LS1 is the most effective for LSGO. MTS-LS1 evaluates each dimension one by one, from the first dimension to the last one. MTS-LS1 is a local search method suitable for separable problems but sensitive to rotations [21].

MTS-LS1 and its variants were used as local search methods in important winner algorithms of the IEEE CEC mentioned above as SHADE-ILS [21], MTS [44], IHDELS [41], MLSHADE-SPA [43], among others. L-BFGS-B is another local search method with great results in LSGO problems. L-BFGS-B utilizes an approximation of the gradient to improve the search and is sensitive to rotation [45]. This method is used in IHDELS [41] and SHADE-ILS [21].

A recent and promising alternative to deal with increased dimensionality is mapping the optimization process to a more straightforward alternative space. Based on this premise, [23], and [24] developed algorithms according to the following idea: while the global optimization problem is defined in ample dimensional space, the decision of whether to start or not a local search occurs through a clustering technique based on a low-dimensional space. A limitation of this approach is that the low-dimensional space is based on feature engineering, which is associated with a problem-specific geometric characteristic of every sampled solution. [18] developed the C-MDE, a clustering-memetic DE that uses Random Projection (RP) to reduce the dimensionality of the LSGO problems. Random Projections are random linear maps sampled from suitable distributions, which approximately preserve certain geometrical invariants. In other words, it is not problem-dependent. However, C-MDE fails to deal appropriately with the loss of information caused by dimension reduction. Aside from that, only 100 dimensions were tested, which is not enough compared to other tests conducted using state-of-the-art algorithms.

Recently, [46] proposed a generic framework of iterative Cooperative Coevolution with evolutions in two spaces (BICCA). In the pattern space, variable interaction patterns are continuously evaluated by CC. The patterns are evolved in the pattern space by using a pattern discovery engine to analyze the interaction of the variables. Cooperative Coevolution and global search are performed adaptively in the search space to obtain better fitness. By adopting evolutions and interactions within two spaces, patterns evolve to provide better clusters while individuals evolve to achieve better fitness. Problem decomposition is conducted throughout the optimization process. Experiments on widely used benchmarks show that BICCA achieves competitive performance on optimization problems with up to 10000 dimensions. [47] proposed an EDA based on latent space (LS-EDA), which transforms the multivariate probabilistic EDA model into its principal component latent subspace with lower dimensionality. Dimensions with higher projected values contribute more to the optimization process. LS-EDA can also help to recognize and understand the structure of the problem. The computational cost and population size of LS-EDA can be effectively reduced due to the dimensionality reduction. At the same time, its performance is highly competitive compared with state-of-the-art algorithms for overlapping and non-separable functions.

The following section presents the proposed algorithm named Low-dimensional Space Modeling based Differential Evolution (LSMDE). This algorithm expands on recent research by proposing a dimensionality reduction algorithm for large-scale global optimization without requiring prior knowledge about the search space. LSMDE also deals with the information loss inherent to the dimensionality reduction

process through a soft clustering process based on probability distributions.

III. Proposed Algorithm

In this section, we introduce a **L**ow-dimensional **S**pace **M**odeling based **D**ifferential **E**volution for Large Scale Global Optimization, referred to as LSMDE. The proposed algorithm is a non-decomposition-based technique, as it optimizes the entire search space through an adaptive dimensionality reduction algorithm and uses Gaussian mixture models to search for the optimal solution in the reduced space. LSMDE comprises three main steps: Population Initialization with modified partial opposition-based learning, Dimensionality Reduction with an Adaptive Singular Value Decomposition strategy, and Search Space Exploration, with GM-SHADE.

A. Population Initialization

In general, a good initial population improves the performance of algorithms and can save computational resources during the search process [48]. Random or uniform population initialization are among the most common techniques. However, these techniques are not recommended for large-scale problems [49]. Opposition-based Learning (OBL) has attracted interest in the past decade, mainly because of its wide use in soft computing algorithms [48]. The computational concept of opposition learning was inspired by the concept of opposite number, defined as follows:

Definition III.1 (Opposite point in the d-space)**.** Let $\vec{x}(x_1, ..., x_d)$ be a point in d-dimensional space and $x_j \in [a_j, b_j], j = 1, 2, ..., d$. The opposite of $\vec{x}$ is defined by $\vec{z}(z_1, ..., z_d)$ as follows:

$$z_j = a_j + b_j - x_j \tag{1}$$

According to [48], searching for optimal solutions considering randomness and its opposite provides a higher probability of finding the promising regions since the search can be redirected to more favorable regions in opposite directions. LSMDE uses a modified partial opposition-based learning inspired by [50] with three different opposition operators for each dimension of the problem, as defined below.

Definition III.2 (LSMDE Opposition Strategy)**.** Let $\vec{x} = (x_1, ..., x_d)$ be a point in d-dimensional space and $x_j \in [a_j, b_j], j = 1, 2, ..., d$. The opposite of $\vec{x}$ is defined by $\vec{z} = (z_1, ..., z_d)$ where each z_j is selected randomly among three opposition operators, as follows :

$$z_j = \begin{cases} a_j + b_j - x_j \\ x_j + a_j - ((a_j + b_j)/2) \mod (b_j - a_j) \\ random(a_j + b_j - x_j, a_j + b_j/2) \end{cases} \tag{2}$$

Generally, traditional opposition schemes calculate the opposition for all variables of a candidate solution using the same opposition operator [48]. This technique limits the diversity of opposition points since the "opposed space" always follows the same formation rule. The random addition of opposition operators allows the construction of opposite points with more significant variability in each dimension, allowing exploring regions of the high-dimensional search space that would be difficult to reach only through an evolutionary algorithm. Therefore, the goal is to generate an initial population with high diversity that allows for more efficient exploration of the d-dimensional search space in the following optimization steps.

Given a minimization problem, a population of individuals $\vec{x}_i \in P$ generated randomly and its opposite population $\vec{z}_i \in Z$ generated according to Equation 2, each new individual $\vec{x}_i$ of the new initial population P is formed by:

$$\vec{x}_i = \min(f(\vec{x}_i), f(\vec{z}_i)) \mid \forall \vec{x}_i, \vec{z}_i \in (P \cup Z) \text{ and } i = 1, 2, ..., n \tag{3}$$

where f is a fitness function and n is the number of individuals in the population. In other words, the initial population is formed by the best individuals between the random population P and the opposite population Z. Before starting the optimization step, LSMDE reduces the dimensionality of the initial generated population P in order to reduce the complexity of the overall Large Scale Optimization Problem.

B. Dimensionality Reduction by Adaptive Singular Value Decomposition

Given any population P is a subset of the search space, LSMDE represents this subset as a matrix $P^{n \times d}$ where n is the number of individuals in the population and d is the dimensionality of the problem. According to Singular Value Decomposition (SVD) [51], any matrix $P^{n \times d}$ can be decomposed into three matrices U, S, V, illustrated in Figure 1, where V^T is the transpose of matrix V as follows:

$$P = U \cdot S \cdot V^T \tag{4}$$

- S is a $d \times d$ diagonal matrix with $s_{11} \geq s_{22} \geq ... s_{dd} \geq 0$ called the singular values.
- U is a $n \times d$ matrix called the left singular vectors or eigensamples $U^T \cdot U = I$
- V is a $d \times d$ matrix call the right singular vectors or eigenfeatures $V \cdot V^T = V^T \cdot V = I$

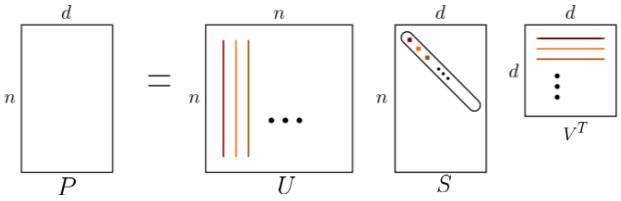

Fig. 1. Matrix decomposition

We can rewrite Equation 4 as follows:

$$P = \Sigma_{i=1}^{\min(n,d)} s_i \cdot \vec{u}_i \cdot \vec{v}_i^T \tag{5}$$

where s_i is the ith singular value and $\vec{u}_i, \vec{v}_i^T$ are the corresponding left singular vectors and right transpose singular vectors. In other words, the SVD expresses P as a non-negative linear combination of $\min(n, d)$ rank-1 matrices, with the singular values providing the multipliers and the outer products of the left and right singular vectors providing the rank-1 matrices. As the SVD of P represents the subspace P as a multiplication of matrices ordered by importance, we can keep only the most important k singular vectors in order to obtain a projection $U \cdot S \in \mathbb{R}^k$ from subspace $P \in \mathbb{R}^d$ where $k < d$. The general idea to reduce

the dimensionality of the matrix P is to keep only the first top k terms on Equation 5 in order to generate a low-rank approximation $\widetilde{P}$ similar to the original P [51] as defined below:

$$\widetilde{P} = \Sigma_{i=1}^{k} s_i \cdot \vec{u}_i \cdot \vec{v}_i^T \text{ or } \widetilde{P} = U_k \cdot S_k \cdot V_k^T \quad (6)$$

Matrix $\Psi = U_k S_k$ gives low dimensional representations of the search space P from its k principal components scores (PC) and matrix V_k^T can be interpreted as a reconstruction matrix that projects these low dimensional points back into the approximate high dimensional space $\widetilde{P}$ (Figure 2).

In order for the Ψ subspace to be used in a population-based approach, it is necessary to develop a way to update the low-dimensional subspace without having to reconstruct it. This feature is important so that mutation, crossover and selection operators can generate new low-dimensional candidate solutions without losing their neighborhood relationship.

For an approximate subspace $\tilde{P}$, the three decomposed matrices U_k, S_k and V_k are computed first. However, when a new individual is added to the search space through a population-based approach, it is not necessary to recalculate the low-dimensional subspace Ψ from scratch. We can use the concept of *Folding-in* to build $\Psi^{(t+i)}$ as an incremental process [52], [53].

Let $U_k S_k V_k^T$ be the matrice that makes up the low-ranking approximation of the subspace $P \in \mathbb{R}^{n \times d}$, given a new individual $\vec{x}_i \in \mathbb{R}^{n \times d}$ to be projected onto a k-dimensional space Ψ, we have $\vec{u} = \vec{x}_i \cdot V_k \cdot S_k^{-1}$. The projection $\vec{u} \in \mathbb{R}^{n \times k}$ is merged into the existing SVD by adding to the bottom of the matrix U_k resulting in a matrix $U_k^{(t+1)} \in \mathbb{R}^{(n+1) \times k}$. This new matrix can be used to compute $\Psi^{(t+1)} = U_k^{(t+1)} \cdot S_k$ for the next generation of the population-based approach. Figure 3 illustrates the projection of a new individual $\vec{x}_i$ into Ψ.

The best value of k to minimize the difference between P and $\widetilde{P}$ can be described by Johnson-Lindenstrauss lemma (JLL) [54], as described below:

Lemma III.1 (Johnson-Lindenstraus)**.** *For any $0 < \epsilon < 1$ and integer n, let k be a positive integer such that $k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n$. Then for any set P of n points in $\mathbb{R}^d$, there is a map $f : \mathbb{R}^d \to \mathbb{R}^k$ such that for all $\vec{x}, \vec{y} \in P$,*

$$(1-\epsilon) \left\|\vec{x} - \vec{y}\right\|^2 \leq \left\|f(\vec{x}) - f(\vec{y})\right\|^2 \leq (1-\epsilon) \left\|\vec{x} - \vec{y}\right\|^2 .$$

Furthermore, this map can be found in randomized polynomial time.

LSMDE uses the JLL to infer k for dimensionality reduction by Singular Value Decomposition in order to preserve the distances of individuals by a factor of $(1 \pm \epsilon)$. By not making assumptions about the topology of the search space, the method is more flexible and can be applied to different search spaces. Based on JLL, it is possible to develop an adaptive ϕ function that maps each high dimensional element $\vec{x}_i \in P$ into its corresponding element $\vec{\psi}_i \in \Psi$ and an inverse function ϕ' that projects these low dimensional individuals back into the high dimensional space such that:

$$\phi(\vec{x}_i) = \vec{\psi}_i,\ \phi'(\vec{\psi}_i) \simeq \vec{x}_i \mid f(\vec{x}_i) \simeq f(\phi'(\vec{\psi}_i))\ ,\ \forall i \in [1, n] \quad (7)$$

$$\phi'(\vec{\psi}_i) = \vec{\psi}_i \cdot V_k^T \quad (8)$$

f is the fitness function for the optimization problem and V_k^T is a reconstruction matrix.

From the concepts presented above, we can develop a function $\phi : \mathbb{R}^d \to \mathbb{R}^k$ that maps candidate solutions from a high-dimensional space P to a space of low dimensionality Ψ. Similarly, we can develop a function $\phi' : \mathbb{R}^k \to \mathbb{R}^d$ such that each individual $\vec{\psi} \in (\Psi \subset \mathbb{R}^k)$ can be mapped back to the individual $\vec{x} \in (P \subset \mathbb{R}^d)$ as the Algorithm 1 and 2 below:

Algorithm 1: Adaptive SVD ϕ

Data: P, n, ϵ
Result: $\widetilde{P}$
1. $k \leftarrow 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \log n$;
2. $U_k, S_k, V_k \leftarrow SVD(P)$;
3. $\Psi \leftarrow (U_k S_k)$;
4. **Return:** Ψ, V_k^T ;

Algorithm 2: Inverse Adaptive SVD ϕ'

Data: Ψ, V_k^T
Result: $\widetilde{P}$
1. $\widetilde{P} \leftarrow \Psi V_k^T$;
2. **Return:** $\widetilde{P}$;

The computation complexity of the other LSMDE steps is asymptotically upper bound by the complexity of the space transformation process defined in Algorithms 1 and 2 since these transformations are responsible for the highest computational cost. According to [55], the computation complexity of the SVD method represented by Algorithm 1, line 2 is $O(k^2 n)$. The matrix multiplication complexity for generating the low dimensional representations of the search space Ψ is also $O(k^2 n)$. In Algorithm 2, line 1, the complexity of projecting a low-dimensional population back to high-dimensional space is $O(knd)$. To summarize, the complexity of converting a population to low dimensionality and back to high dimensionality is $O(k^2 n + knd)$.

Success-History Based Parameter Adaptation for Differential Evolution (SHADE) is one of the most successful differential evolution algorithms [56] and some of its variants are among the best algorithms in important competitions [21], [57]. Therefore, LSMDE applies the SHADE, hybridized with a Gaussian Mixture Model with a variational bayesian estimation mechanism in the low dimensionality space to discover regions of optimality that can be reconverted to the original space. This hybridization is named GM-SHADE.

C. Gaussian Mixture SHADE (GM-SHADE)

One of the key tasks in the application of mixture models is the determination of a suitable number of clusters. GM-SHADE estimates the model parameters and detects the number of clusters automatically as part of the variational estimation procedure using a parameter based on a finite mixture model with Dirichlet process [58]. The Dirichlet Process is a stochastic process used in Bayesian nonparametric to cluster data without specifying the number of clusters in advance.

To perform the search in a low-dimensional space, we created the GM-SHADE, an algorithm that hybridizes the SHADE algorithm and Bayesian Gaussian Mixture Models

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2022.3227440

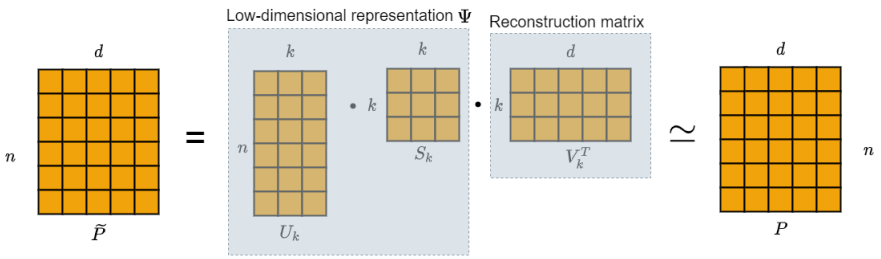

Fig. 2. Illustration of a low-dimensional representation from a d-dimensional random P matrix to a k-dimensional Ψ matrix.

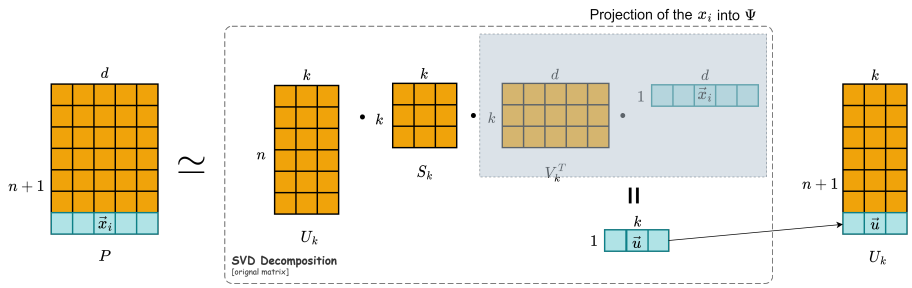

Fig. 3. Projection of a new high dimensional individual $\vec{x}_i$ through the Low-rank approximation $\phi(\vec{x}_i \in P)$ (folding-in)

to guide the search process for promising regions in a low-dimensional search space. This algorithm has a parameter adaptation based on a successful history inspired by the SHADE-ILS algorithm [21], that is, crossover rate (CR) and a scaling factor (F) are dynamically updated following the same rules defined in [21].

GM-SHADE searches for promising regions in the low-dimensional space Ψ, framing it into clusters that are represented by a function composed of several Gaussians, each identified by $\lambda \in 1, ..., \Lambda$, where Λ is the current number of clusters. The cluster coverage is determined by a Gaussian λ, composed by a mean individual $\vec{\mu}$, a covariance Υ and a mixing probability π. The goal is to ensure that each Gaussian fits all candidate solutions $\vec{\psi}_i$ belonging to each cluster. The assimilation process from a candidate solution $\vec{\psi}_i$ to a Gaussian mixture is a Gaussian Density Function:

$$N(\vec{\psi}_i|\vec{\mu}, \Upsilon) = \frac{1}{(2\pi)^{k/2}|\Upsilon|^{1/2}} * e^{-\frac{1}{2}(\vec{\psi}_i - \vec{\mu})^T \Upsilon^{-1} (\vec{\psi}_i - \vec{\mu})} \quad (9)$$

$\vec{\psi}_i$ represents an individual in the low-dimensional space, k is the number of dimensions of $\vec{\psi}_i$, $\vec{\mu}$ is the mean and Υ is the covariance. Using the variational expectation-maximization algorithm (Variational EM) [58], GM-SHADE finds the best parameters $\theta = [\pi_\lambda, \vec{\mu}_\lambda, \Upsilon_\lambda]$ for each Gaussian λ. The mixing coefficient π_λ represents the overall probability of observing an individual $\vec{\psi}_i$ that comes from Gaussian λ.

Dirichlet process mixture models (DPMMs) provide a non-parametric Bayesian framework to describe distributions over mixture models with an infinite number of mixture components (clusters). A Dirichlet process (DP), parameterized by a base distribution G_0 and a concentration parameter α, is used as a prior over the distribution G of the clusters. For a population Ψ, mixture component parameters θ, and a parameterized distribution H, the DPMM can be written as Equation 10:

$$\begin{aligned} G|\alpha, G_0 &\sim DP(\alpha, G_0) \\ \theta_i|G &\sim G \\ \vec{\psi}_i|\theta_i &\sim H(\theta_i) \end{aligned} \quad (10)$$

Dirichlet process mixture models can be implemented as a Gaussian mixture model (GMM) in which all parameters, including Λ, are inferred from Ψ [59]. In GMM, Gaussians with high π values represent regions where more individuals ψ are observed and are promising regions. Figure 4 shows the number of clustering being updated according to the population Ψ.

In this work, we consider that a Gaussian λ is promising if:

$$\pi_\lambda \geq 0.25(1/\Lambda) + (1/\Lambda) \quad (11)$$

where Λ is the current number of clusters inferred by DP to the current population Ψ. The promising Gaussian defined by $\theta = [\pi_p, \vec{\mu}_p, \Upsilon_p]$ represents the region of search space where there are higher chances of finding significant solutions to the problem and $\vec{\mu}_p$ is the average individual that represents the promising Gaussian. In each generation, a mutant vector $\vec{v}_i$ is generated from a member $\vec{\psi}_i$ of the current low-dimensional population according to the Equation as follows:

$$\vec{v}_i = \vec{\psi}_i + F_i \cdot (\vec{\mu}_p - \vec{\psi}_i) + F_i \cdot (\vec{\psi}_{r1} - \vec{\psi}_{r2}) \quad (12)$$

$\vec{\mu}_p$ is the mean individual of the most promising Gaussian in the current generation, and the scaling factor F_i is generated from the Cauchy distribution as defined in [21]. The indexes $r1$ and $r2$ indicate that the individuals $\vec{\psi}_{r1}$ and $\vec{\psi}_{r2}$ are chosen randomly from the population Ψ. After generating the mutant vector $\vec{v}_i$ it is crossed with the parent $\vec{\psi}_i$ in order to generate trial vector $\vec{\rho}_i$ as follows:

$$\vec{\rho}_{i,d} = \begin{cases} \vec{v}_{i,d} & \text{If rand[0,1)} \leq CR_i \\ \vec{\psi}_{i,d} & \text{Otherwise} \end{cases} \quad (13)$$

$rand[0,1)$ denotes a uniformly selected random number from $[0,1)$, $\vec{\rho}_{i,d}$ is the d-th dimension of the i-th trial vector $\vec{\rho}$ and $CR_i \in [0,1]$ is the crossover rate of individual $\vec{\psi}_i$.

After all of the trial vectors $\vec{\rho}_i$ have been generated, a selection process determines the survivors for the next low-dimensional population $\Psi^{(t+1)}$. The selection operator compares each $\vec{\psi}_i$ against its corresponding trial vector $\vec{\rho}_i$, keeping the better vector in the population. In a minimization problem, the crossover is defined as follows:

$$\vec{\psi}_i = \begin{cases} \vec{\rho}_i & \text{If } f(\phi'(\vec{\rho}_i)) \leq f(\phi'(\vec{\psi}_i)) \\ \vec{\psi}_i & \text{Otherwise} \end{cases} \quad (14)$$

When a stopping condition is reached, GM-SHADE returns the $\theta = [\pi_p, \vec{\mu}_p, \Upsilon_p]$ of the promising Gaussians from the current population Ψ. In a multimodal problem, more than one Gaussian may be promising. Figure 4 illustrates the clustering of the search space by Bayesian Gaussian Mixture models as the Ψ population is updated in each generation. The average individual $\vec{\mu}_p$ is returned to the original search space using the inverse of the adaptive SVD function ϕ' as follows:

$$\phi'(\vec{\mu}_p) = \vec{\mu}_p \cdot V_k^T \quad (15)$$

Promising Gaussian $\theta = [\pi_p, \vec{\mu}_p, \Upsilon_p]$ reconverted in a promising high dimensional neighborhood can be explored by a specialized local search algorithm aiming to find the optimum solution for the LSGO problem in a limited region of the approximated original search space (Figure 5).

As [21], GM-SHADE uses a combination of two local search algorithms to explore the promising high-dimension neighborhood: MTS LS1 [44], specially designed for LSGO and L-BFGS-B [45] that uses an approximation of the gradient.

IV. Experiments and Results

In order to evaluate the performance of our proposal, we adopted the most recent test suite for LSGO problems held at 2013 *IEEE Congress on Evolutionary Computation (CEC'2013)* [25], adopting new evaluation criteria held at 2019 *IEEE Congress on Evolutionary Computation (CEC'2019)* [26]. CEC'2013 benchmark comprises 15 minimization functions with 1000 dimensions, except for f_{14} and f_{15}, which are overlapping functions where $d = 905$. The functions are divided into 5 categories: Fully-separable functions ($f_1 - f_3$), Functions with a separable subcomponent ($f_4 - f_7$), Functions with no separable subcomponent ($f_8 - f_{11}$), Overlapping functions ($f_{12} - f_{14}$) and Non-separable functions (f_{15}). In addition, tThe CEC'2013 benchmark improved earlier versions by including nonuniform subcomponent sizes, Imbalance in the contribution of subcomponents, functions with overlapping subcomponents, new transformations such as symmetry breaking, ill-conditioning, and local irregularities [3], [25]

The IEEE CEC'2013 benchmark suite was designed to provide a suitable evaluation platform for testing and comparing large-scale optimization algorithms [25]. To that end, the CEC'2013 benchmark suite successfully represents the nature of a variety of real-world problems and builds a scalable set of benchmark functions to promote research in the field of large-scale global optimization [60].

As an example, it is argued that subcomponent interaction is commonplace in many real-world problems [61]. For example, each component of a supply-chain problem is called a silo, and most are multi-silo problems with interaction between silos [62]. CEC'2013 contemplates this class of problems sharing decision variables between subcomponents (overlapping decision vectors or coupling variables). Interconnected components are prevalent in many engineering optimization problems, such as multidisciplinary design optimization (MDO) [63] where a problem should be analyzed from various engineering aspects. For instance, the design of an airplane wing may have an aerodynamic and a structural design aspect, both of which can impose some constraints on the other [64]. Another example is an automotive design process where a power train sequentially follows structural, chassis, and interior design. [64]

Table I presents some features of the CEC'2013 benchmark.

TABLE I
SUMMARY OF THE CEC 2013 LSGO BENCHMARK FUNCTIONS

	Function	Properties	Search Range
f_1	Elliptic Function	Unimodal	$[-100,100]^D$
f_2	Rastrigin Function	Multimodal	$[-5,5]^D$
f_3	Ackley Function	Multimodal	$[-32,32]^D$
f_4	Elliptic Function	Unimodal	$[-100,100]^D$
f_5	Rastrigin Function	Multimodal	$[-5,5]^D$
f_6	Ackley Function	Multimodal	$[-32,32]^D$
f_7	Schwefels Problem 1.2	Multimodal	$[-100,100]^D$
f_8	Elliptic Function	Unimodal	$[-100,100]^D$
f_9	Rastrigin Function	Multimodal	$[-5,5]^D$
f_{10}	Ackley Function	Multimodal	$[-32,32]^D$
f_{11}	Schwefels Problem 1.2	Unimodal	$[-100,100]^D$
f_{12}	Rosenbrock's Function	Multimodal	$[-100,100]^D$
f_{13}	Schwefels Function	Unimodal	$[-100,100]^D$
f_{14}	Schwefels Function	Unimodal	$[-100,100]^D$
f_{15}	Schwefels Problem 1.2	Unimodal	$[-100,100]^D$

Seven state-of-the-art metaheuristic approaches: BICCA [46], CC-CMA-ES [31], C-DEEPSO [35], GL-SHADE [20], IHDELS [41], MLSHADE-SPA [43] and SHADE-ILS [21] are selected to participate in comparison against LSMDE.

Competing algorithms were chosen according to their rankings in the past *IEEE CEC Special Sessions and Competitions on Large-Scale Global Optimization* and their similarities to LSMDE. For example, SHADE-ILS uses the same Differential Evolution algorithm (DE) that is the basis for the DE of the proposed algorithm. In addition, SHADE-ILS was the best algorithm for the *IEEE CEC'2018 Special Session and Competition on Large-Scale Global Optimization*. MLSHADE-SPA, GL-SHADE, and IHDELS are some algorithms that use a similar DE to SHADE-ILS. They also have promising performance in a different function category from the CEC'2013 benchmark. BICCA was selected for an exploratory search strategy in an alternative search space to the original search space. This feature is similar to LSMDE, although it does not use a dimensionality reduction strategy. CC-CMA-ES and DEEPSO were selected because they are adaptations for LSGO of two algorithms that traditionally have good performance in continuous optimization problems: Covariance Matrix Adaptation Evolution Strategy (CMA-ES) and Particle Swarm Optimization (PSO).

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2022.3227440

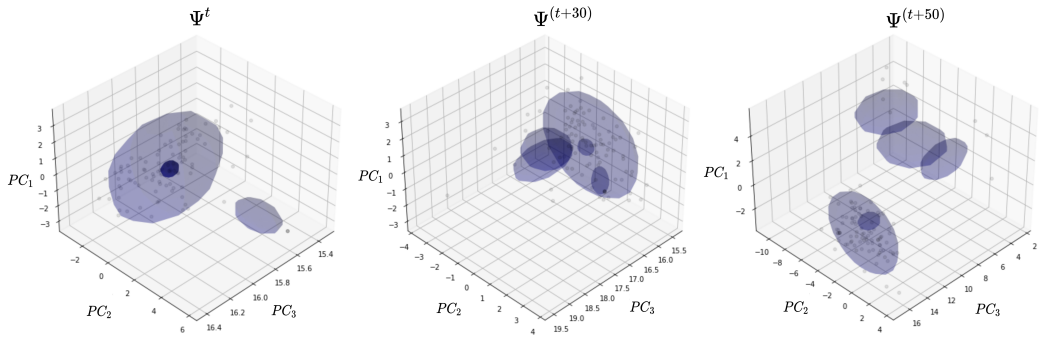

Fig. 4. Gaussian Mixture Models finding promising regions in the Alpine N1 low-dimensional search space. The number of clusters is automatically inferred by DP. In this case the number of clusters initially defined (population Ψ^t) does not match the true generative distribution of the next populations; therefore, the Bayesian Gaussian mixture model fits the number of clusters according to the current population distribution.

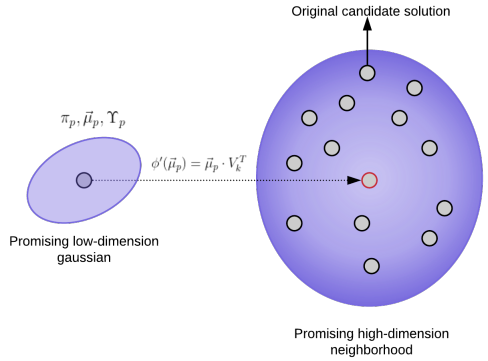

Fig. 5. Illustration of a space transformation from a promising Gaussian in Ψ to a promising region in $\widetilde{P}$.

Based on the criteria defined in [26], three experiments were performed with different maximum numbers of objective function evaluations (FE): 1.2×10^5 evaluations, 6×10^5 evaluations and 3×10^6 evaluations. Twenty-five independent executions per function were carried out for each experiment and algorithm. Running averages of each competing algorithm used in this work are described in recent works, gathered in *Special Sessions and Competition on Large-Scale Global Optimization* and available at *www.tflsgo.org*. The control parameters adopted by the competing algorithms are described in their respective papers, and the control parameters adopted by LSMDE are shown in Table II. Tolerance ϵ specifies the allowed distortion when constructing the low-dimensional space Ψ. Higher ϵ values indicate a greater tolerance for distortion and lower k values. However, as information loss increases, the optimization procedure becomes more difficult. Population size was determined in the same way as in competing shade-based algorithms, and ϵ was determined experimentally.

TABLE II
PARAMETERS USED IN LSMDE

Parameter	Value	Description
n	100	Size of the Population
d	1000	Dimension of the problem
ϵ	0.03	Johnson-Lindenstraus tolerance error

A. Performance measurement

The performance comparison is carried out using a method indicated by *IEEE Task Force on Large-Scale Global Optimization* based on Formula One car racing (Formula One criterion). In particular, this method was used in the recent CEC LSGO competition 2019 [65]. This process of optimizing a function is analogous to a race, such that the competitor (algorithm) that comes in first (best average performance) receives 25 points, second place receives 18 points, the third receives 15 points, and so on. Table III presents the summary of the scores in competition with eight competitors (the number of competitors in this work).

TABLE III
SUMMARY OF SCORES

Position	score
1^{st}	25
2^{nd}	18
3^{nd}	15
4^{th}	12
5^{th}	10
6^{th}	8
7^{th}	6
8^{th}	4

Table IV emphasizes the maximum score that can be achieved by a competitor based on the number of functions contained in each of the categories of functions, allowing us to do a deeper analysis of the performance of the algorithms for each category. Although the maximum score in an experiment is 375, as the number of functions per category is not equal, the maximum scores per category are also not equal.

TABLE IV
SUMMARY OF MAXIMUM SCORES PER CATEGORY OF FUNCTION

Category	nº of functions	Max
Fully-separable Functions	3	75
Functions with a separable subcomponent	4	100
Functions with no separable subcomponents	4	100
Overlapping Functions	3	75
Non-separable Functions	1	25

Figure 6 presents the average cumulative score of the competing algorithms for the three experiments proposed by

the *IEEE Task Force on Large-Scale Global Optimization*. It can be observed that LSMDE has the best overall average score, which indicates a possible superiority over the other algorithms. However, it is essential to observe the performance of each algorithm on each category of function. This analysis allows us to identify which types of LSGO problems the algorithms performed better and worse on. To perform this analysis, each experiment was explored separately (1.2×10^5 evaluations, 6×10^5 evaluations, and 3×10^6 evaluations).

Table V shows the average fitness values for each function during the execution of the experiments with the limit of 1.2×10^5, 6×10^5 and 3×10^6 evaluations of the objective function f. Based on Table V, it is possible to observe that the proposed algorithm LSMDE has the best average performance for an experiment with 1.2×10^5 evaluations of the objective function f. Figure 7(a) shows the score of each algorithm based on the Formula One criterion for each category of function for $FE = 1.2 \times 10^5$. The proposed algorithm demonstrates its best performance on functions with a separable component, functions with no separable components, and non-separable functions. LSMDE presented an average score superior to all competing algorithms, including one of the best current algorithms, SHADE-ILS.

LSMDE demonstrated better average performance than the other algorithms for an experiment with 6×10^5 evaluations of the objective function f. Figure 7(b) confirms the superiority of LSMDE on functions with a separable subcomponent and functions with no separable subcomponents. We can also observe a promising performance in overlapping functions for $FE = 6 \times 10^5$.

GL-SHADE algorithm scored higher than other competing algorithms for the experiment with a limit of 3×10^6 evaluations of the objective function f. LSMDE and SHADE-ILS presented the second-best position. Figure 7(c) shows that LSMDE still performs better in functions with a separable subcomponent and functions with no separable subcomponents than GL-SHADE and SHADE-ILS for $FE = 3 \times 10^6$

Figure 6 shows that the proposal has the highest average overall score. According to Figure 8, LSMDE has the best performance in experiments with a maximum of 1.2×10^5 and 6×10^5 evaluations, achieving the third best position in the experiment with a maximum of 3×10^6 evaluations. Figure 8 presents the general score for each experiment with the sum of the scores of each function category.

V. Statistical Analysis

In order to guarantee the statistical confidence of the previous results, we selected the state-of-the-art algorithm, the most recent algorithm among the competitors, and the proposed algorithm (SHADE-ILS, GL-SHADE, LSMDE) to carry out a more in-depth analysis. As their papers only provide the means and standard deviations of each competing algorithm (some papers only provide the means, without the standard deviations) and it is not possible to identify the results of each run, LSMDE, SHADE-ILS and GL-SHADE were run locally to apply the appropriate statistical tests. Initially, we planned to use the multivariate analysis of variance test (MANOVA) with two factors (function and algorithm), but the data provided by running the three algorithms did not pass the homoscedasticity test.

According to [66], the non-parametric Kruskal-Wallis test is recommended for comparing performance among three bio-inspired algorithms. However, the statistical test only indicates whether or not there is a statistical difference among the three compared algorithms. We used two post hoc tests, Conover and Dunn, with Holm correction to determine which algorithm has the best performance [67].

Table VI and VII present the p-values of the LSMDE versus the previous best algorithms for Dunn and Conover post hoc tests, respectively. p-values smaller than 0.05 indicate a statistical difference between the control algorithm and the algorithm compared with the current function. In this case, LSMDE is the control algorithm. p-values in bold represent functions where LSMDE performed statistically superior to its adversary (SHADE-ILS or GL-SHADE). The other p-values represent functions where the LSMDE is statistically equal to or lower than its adversary.

Table VI and VII summarize the results of the comparison of the proposed algorithm LSMDE versus the state-of-the-art algorithm SHADE-ILS and the comparison of LSMDE versus GL-SHADE (one of the most recent algorithms for LSGO problems). These results can be analyzed considering each experiment separately: experiment 1 with 1.2×10^5 evaluations, experiment 2 with 6×10^5 evaluations and experiment 3 with 3×10^6 evaluations.

Based on experiments 1 and 2 (Table VI and VII), we can say that the proposed algorithm is statistically superior to GL-SHADE in 11 or 12 of the 15 functions from the CEC'2013 benchmark. GL-SHADE outperformed LSMDE on 9 functions for experiment 3. However, the proposed algorithm performed better on partially separable functions, outperforming GL-SHADE on 5 of the eight functions available in this category (Figure 10). Comparing LSMDE and SHADE-ILS shows an improvement of the proposed algorithm concerning the number of statistically better functions in experiments 1 and 2. This improvement is more evident when we analyze the category of partially separable functions ($f_4 - f_{11}$) as observed in Figure 10. In experiment 3, it is observed that LSMDE is statistically similar to, or superior, in 7 functions and statistically inferior to SHADE-ILS in 8 functions. Figure 10 shows that LSMDE achieved better performance in 4 functions for partially separable problems in experiment 3, according to the Conover post hoc test.

The convergence curves of LSMDE confirm its advantages over the best competitors. Algorithms are on a logarithmic scale which helps to see more clearly how fast algorithms converge across the calculation lifecycle, and all algorithms start from the same seed for a fair comparison. Based on Figure 9, we can infer that the LSMDE outperforms the competition in tests with less objective function evaluations. A limited evaluation indicates an advantage of the proposed algorithm in situations with limited computational resources. We also observe that LSMDE performs better on partially separable functions ($f_4 - f_{11}$).

Analyses using the Formula One criterion suggested by *IEEE Task Force on Large-Scale Global Optimization* demonstrate that the proposed algorithm performs better than other state-of-the-art algorithms. Nonetheless, the Kruskal-Wallis statistical test enables an evaluation of which function in each competing algorithm performs best, equal, or worse.

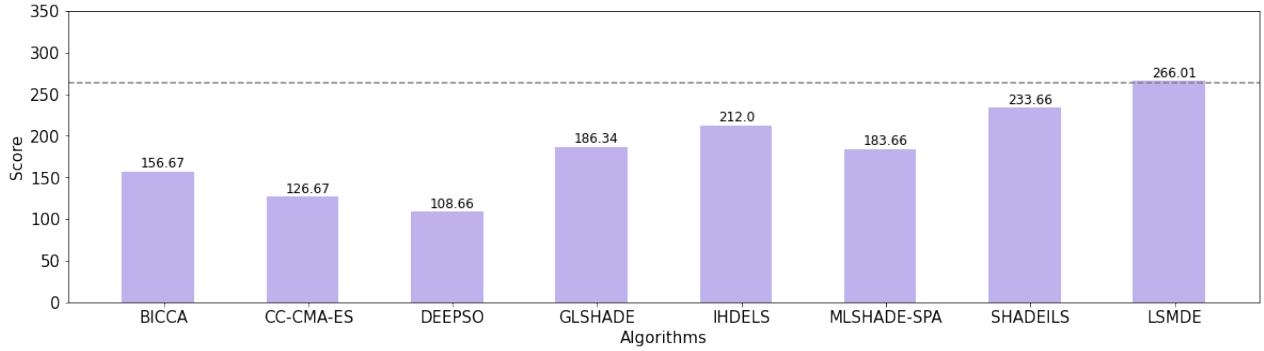

Fig. 6. Overall average score for 8 competitors based on Formula One criterion considering the three experiments. LSMDE has the best overall performance.

Since each function represents a broader range of real-world optimization problems, the statistical test enables the most suitable algorithm for each situation to be chosen. Therefore, the proposed algorithm performs best, especially on partially separable problems. We can also conclude that LSMDE is competitive with the best algorithms used in this work.

VI. Conclusions and Future Work

In this paper, we proposed a new SHADE-based non-decomposition algorithm (LSMDE) to solve Large-Scale Global Optimization problems. LSMDE uses a Singular Value Decomposition algorithm to reduce the $d-$dimensional search space according to the resources of the current population of candidate solutions. The search for promising solutions in low-dimensional space is guided by a SHADE algorithm hybridized with Gaussian Mixture Models. This model allows GM-SHADE to cluster low-dimensional solutions, considering the uncertainty of their mapping to the low-dimensional space and mitigating the loss of information in this process. Candidates for the optimal solution can be mapped back to high-dimensional space through an Inverse SVD algorithm, and their neighborhood region can be properly explored through a local search algorithm. In order to evaluate the LSMDE performance, this algorithm has been tested on the most recent test suite for LSGO problems, IEEE CEC'2013 benchmark functions.

The No Free Lunch Theorem [68] proves mathematically that no algorithm can perform the best on all problems. Nonetheless, according to the experimental result, the proposed algorithm has relevant advantages summarized as follows: (i) LSMDE outperforms many state-of-the-art algorithms for partially separable functions, which may indicate it has better applicability to real-world problems; (ii) LSMDE showed the best overall performance in experiments with limited computational resources (1.2×10^5 and 6×10^5 objective function evaluations); (iii) Its innovative dimensionality reduction method can be adapted and improved for different population-based algorithms; (iv) Its dimensionality reduction method learns from the current population of candidate solutions without any prior knowledge about the search space characteristics, making the LSMDE more flexible and applicable to a variety of high-dimensional problems.

Experiments also revealed some limitations of the proposed algorithm. First, LSMDE performs worse than competing algorithms on simple problems, such as fully separable problems. Second, the convergence curve demonstrates that although performance is enhanced with low computing resources, there is no substantial gain when a large computational resource is available (This problem is observed in other algorithms based on dimensionality reduction as BICCA and LS-EDA).

Some open issues are worthy of further study in future works: determining the most suitable number of individuals for each dimensionality reduction. In addition, it is essential to investigate the effect of dimensionality reduction on the topology of different types of search space and how this can produce insights for algorithm improvement. We also aim to apply the proposed LSMDE in various applications, such as parameter optimization, constrained optimization problems, multi-objective optimization, data mining, and feature selection.

Acknowledgment

This research was financed in part by the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES)* - Finance Code 001 and by a research grant from Science Foundation Ireland (SFI) under grant no. SFI/16/RC/3918 (CONFIRM) and Marie Sklodowska-Curie grant agreement no. 847.577 co-funded by the European Regional Development Fund.

References

[1] K. S. Kim and Y. S. Choi, "An efficient variable interdependency-identification and decomposition by minimizing redundant computations for large-scale global optimization," *Information Sciences*, vol. 513, pp. 289–323, 2020.

[2] I. De Falco, A. Della Cioppa, and G. A. Trunfio, "Investigating surrogate-assisted cooperative coevolution for large-scale global optimization," *Information Sciences*, vol. 482, pp. 1–26, 2019.

[3] M. N. Omidvar, X. Li, and X. Yao, "A review of population-based metaheuristics for large-scale black-box global optimization: Part b," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[4] M. N. Omidvar and X. Li, "Evolutionary large-scale global optimization: an introduction," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2017, pp. 807–827.

[5] D. Tabernik and D. Skočaj, "Deep learning for large-scale traffic-sign detection and recognition," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 4, pp. 1427–1440, 2019.

[6] M. N. Omidvar, X. Li, and X. Yao, "A review of population-based metaheuristics for large-scale black-box global optimization: Part a," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[7] L. Li, W. Fang, Y. Mei, and Q. Wang, "Cooperative coevolution for large-scale global optimization based on fuzzy decomposition," *Soft Computing*, vol. 25, no. 5, pp. 3593–3608, 2021.

[8] C. L. Chowdhary, P. V. Patel, K. J. Kathrotia, M. Attique, K. Perumal, and M. F. Ijaz, "Analytical study of hybrid techniques for image encryption and decryption," *Sensors*, vol. 20, no. 18, p. 5162, 2020.

TABLE V
AVERAGE FITNESS VALUES FOR THE ALGORITHMS ON THE CEC 2013 LSGO FUNCTIONS WITH 1.2×10^5, 6×10^5 AND 3×10^6 EVALUATIONS.

		Algorithms							
Functions	FE's	BICCA	CC-CMA-ES	DEEPSO	GL-SHADE	IHDELS	SHADE-ILS	MLSHADE-SPA	LSMDE
f_1	1.2×10^5	1.03e+10	1.14e+08	2.99e+10	1.78e+05	2.01e+05	**5.12e+04***	6.12e+07	2.15e+06
	6×10^5	7.87e+03	3.99e+04	4.61e+09	3.82e+01	4.70e+02	**3.55e-23***	1.22e-01	3.50e-09
	3×10^6	**0.00e+00***	5.80e-09	1.44e+08	3.74e-23	4.34e-28	2.56e-28	1.94e-22	6.88e-23
f_2	1.2×10^5	2.26e+04	1.40e+03	3.45e+04	**6.84e+02***	4.99e+03	2.54e+03	1.74e+03	2.83e+03
	6×10^5	**1.01e+01***	1.33e+03	2.61e+04	2.35e+01	2.78e+03	1.79e+03	9.13e+01	1.75e+03
	3×10^6	**8.46e-07***	1.33e+03	1.49e+04	7.76e+00	1.32e+03	1.04e+03	7.89e+01	9.56e+02
f_3	1.2×10^5	1.75e+01	**3.78e-02***	2.12e+01	2.00e+01	2.01e+01	2.01e+01	3.54e+00	2.04e+01
	6×10^5	1.92e+00	**0.00e+00***	2.09e+01	2.00e+01	2.01e+01	2.01e+01	6.74e-05	2.01e+01
	3×10^6	7.27e-01	**0.00e+00***	2.04e+01	2.00e+01	2.01e+01	2.01e+01	0.00e+00	2.01e+01
f_4	1.2×10^5	8.16e+10	6.09e+11	4.27e+11	5.63e+10	**2.15e+10***	3.58e+10	6.92e+11	2.38e+10
	6×10^5	8.23e+09	3.53e+10	4.95e+10	3.44e+09	2.36e+09	3.74e+09	5.63e+09	**2.17e+09***
	3×10^6	8.85e+08	2.19e+09	4.77e+09	**3.01e+07***	3.04e+08	3.01e+08	6.90e+08	1.96e+08
f_5	1.2×10^5	6.31e+06	7.28e+14	1.56e+07	4.35e+06	1.24e+07	2.35e+06	1.09e+07	**2.26e+06***
	6×10^5	3.29e+06	7.28e+14	1.46e+07	2.67e+06	1.02e+07	2.10e+06	2.58e+06	**1.75e+06***
	3×10^6	2.58e+06	7.28e+14	1.45e+07	2.23e+06	9.59e+06	1.33e+06	1.80e+06	**1.15e+06***
f_6	1.2×10^5	**4.82e+05***	7.54e+05	1.05e+06	1.05e+06	1.05e+06	1.05e+06	8.38e+05	1.01e+06
	6×10^5	1.87e+05	6.30e+05	1.04e+06	1.05e+06	1.03e+06	1.04e+06	**1.60e+03***	1.00e+06
	3×10^6	1.46e+05	5.87e+05	1.02e+06	1.03e+06	1.03e+06	1.03e+06	**1.40e+03***	9.98e+05
f_7	1.2×10^5	2.27e+09	5.81e+09	3.50e+09	1.50e+09	3.09e+08	3.68e+08	6.27e+09	**1.52e+08***
	6×10^5	5.34e+07	1.42e+09	2.40e+08	2.80e+07	1.10e+07	**1.54e+06***	1.89e+08	1.05e+07
	3×10^6	1.82e+05	7.44e+06	1.54e+07	**2.37e+00***	3.46e+04	2.24e+02	5.31e+04	3.91e+04
f_8	1.2×10^5	1.87e+15	2.93e+16	8.33e+14	4.05e+14	2.09e+14	2.36e+14	2.46e+16	**7.11e+11***
	6×10^5	3.96e+13	2.30e+15	1.68e+14	1.10e+13	2.22e+13	1.43e+13	4.26e+13	**1.36e+11***
	3×10^6	3.78e+12	3.88e+14	5.42e+12	1.11e+11	1.36e+12	5.99e+11	9.77e+12	**2.68e+09***
f_9	1.2×10^5	5.73e+08	7.36e+08	1.03e+09	2.55e+09	7.25e+08	2.87e+08	8.23e+08	**2.62e+08***
	6×10^5	3.19e+08	4.48e+08	9.29e+09	2.40e+09	6.96e+08	2.49e+08	2.15e+08	**2.13e+08***
	3×10^6	2.18e+08	3.71e+08	9.17e+08	2.36e+09	6.74e+08	1.58e+08	1.61e+08	**1.48e+08***
f_{10}	1.2×10^5	**4.00e+06***	2.68e+07	9.41e+07	9.38e+07	9.43e+07	9.39e+07	3.11e+07	9.21e+07
	6×10^5	1.89e+06	4.49e+06	9.22e+07	9.27e+07	9.31e+07	9.32e+07	**1.11e+03***	9.08e+07
	3×10^6	1.24e+06	7.55e+05	9.07e+07	9.17e+07	9.16e+07	9.26e+07	**6.56e+02***	9.06e+07
f_{11}	1.2×10^5	8.48e+10	6.04e+11	7.52e+11	9.44e+11	9.55e+09	5.58e+09	8.67e+11	**4.30e+09***
	6×10^5	8.59e+08	5.15e+10	8.03e+09	9.27e+11	4.51e+08	**1.30e+08***	1.68e+09	6.96e+08
	3×10^6	2.85e+07	1.58e+08	5.60e+08	9.27e+11	1.07e+07	**5.39e+05***	4.04e+07	1.15e+07
f_{12}	1.2×10^5	1.20e+11	6.92e+03	7.80e+11	2.42e+04	**2.07e+03***	2.64e+03	1.19e+07	4.43e+04
	6×10^5	1.67e+04	6.17e+03	1.55e+11	9.60e+02	1.44e+03	1.77e+03	1.02e+03	**6.68e+02***
	3×10^6	1.40e+03	1.27e+03	1.54e+10	**3.19e-01***	3.77e+02	6.49e+01	1.04e+02	3.25e+02
f_{13}	1.2×10^5	2.11e+10	5.44e+10	1.82e+11	2.69e+10	1.06e+10	1.36e+10	4.29e+10	**8.32e+09***
	6×10^5	8.24e+08	1.87e+10	1.29e+10	2.73e+09	7.29e+08	**5.60e+08***	4.94e+09	1.10e+09
	3×10^6	09e+07	6.69e+08	8.75e+08	**3.98e+04***	3.80e+06	1.07e+06	7.21e+07	7.12e+06
f_{14}	1.2×10^5	2.36e+11	8.18e+11	2.18e+12	3.70e+11	8.96e+10	1.84e+11	1.06e+12	**5.99e+10***
	6×10^5	4.68e+08	1.81e+11	1.63e+11	1.42e+10	**1.67e+08***	4.91e+08	2.56e+10	1.11e+09
	3×10^6	4.27e+07	7.10e+07	4.33e+08	**4.79e+06***	1.58e+07	7.63e+06	1.52e+07	2.54e+07
f_{15}	1.2×10^5	9.09e+08	1.20e+08	6.60e+12	1.12e+08	5.34e+07	8.86e+07	5.66e+08	**4.23e+07***
	6×10^5	2.21e+07	4.24e+07	3.00e+07	3.24e+07	**1.49e+07**	1.69e+07	4.88e+07	2.48e+07
	3×10^6	3.16e+06	3.03e+07	7.04e+06	1.29e+06	2.81e+06	**8.68e+05***	2.76e+07	6.16e+06
Best	1.2×10^5	2	1	0	1	2	1	0	**8**
	6×10^5	1	1	0	0	2	4	2	**5**
	3×10^6	1	1	0	**5**	0	2	3	3

* Best result found for this function.

[9] J. Tamang, J. D. D. Nkapkop, M. F. Ijaz, P. K. Prasad, N. Tsafack, A. Saha, J. Kengne, and Y. Son, "Dynamical properties of ion-acoustic waves in space plasma and its application to image encryption," *IEEE Access*, vol. 9, pp. 18 762–18 782, 2021.

[10] X. Luan, B. De Schutter, L. Meng, and F. Corman, "Decomposition and distributed optimization of real-time traffic management for large-scale railway networks," *Transportation Research Part B: Methodological*, vol. 141, pp. 72–97, 2020.

[11] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, and Q. Wu, "An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization," *Swarm and Evolutionary Computation*, vol. 60, p. 100789, 2021.

[12] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimisation problems," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 2, pp. 150–194, 2013.

[13] F. Caraffini, F. Neri, and L. Picinali, "An analysis on separability for memetic computing automatic design," *Information Sciences*, vol. 265, pp. 1–22, 2014.

[14] S. Jafari, T. Kapitaniak, K. Rajagopal, V.-T. Pham, and F. E. Alsaadi,

TABLE VI
DUNN'S TEST RESULTS, p-VALUES OF LSMDE VERSUS SHADE-ILS AND GL-SHADE ON THE CEC 2013 BENCHMARK FUNCTIONS FOR 1000 DIMENSIONS. SYMBOLS '+','-' AND '=' DENOTE THE LSMDE IS BETTER THAN, WORSE THAN, OR SIMILAR TO THE COMPARED ALGORITHM, RESPECTIVELY.

	Experiment 1: 1.2×10^5		Experiment 2: 6×10^5		Experiment 3: 3×10^6	
Functions	SHADE-ILS	GL-SHADE	SHADE-ILS	GL-SHADE	SHADE-ILS	GL-SHADE
f_1	5.63e-30 (-)	1.38e-08 (-)	1.74e-08 (-)	**1.74e-08** (+)	6.76e-11 (-)	**5.49e-05** (+)
f_2	1.73e-01 (=)	3.72e-15 (-)	**2.48e-05** (+)	1.38e-10 (-)	**4.54e-06** (+)	4.62e-10 (-)
f_3	2.81e-02 (-)	6.77e-22 (-)	1.17e-01 (=)	1.39e-20 (-)	3.00e-01 (=)	3.68e-20 (-)
f_4	7.65e-01 (=)	**1.04e-11** (+)	7.04e-02 (=)	**2.49e-04** (+)	2.91e-02 (-)	2.13e-21 (-)
f_5	6.76e-02 (=)	**2.37e-14** (+)	**1.06e-02** (+)	**2.78e-16** (+)	2.68e-01 (=)	**8.01e-19** (+)
f_6	**9.36e-14** (+)	**6.08e-18** (+)	**6.96e-10** (+)	**1.32e-27** (+)	**8.05e-16** (+)	**2.15e-19** (+)
f_7	**7.58e-08** (+)	**6.88e-29** (+)	7.61e-09 (-)	**3.07e-07** (+)	4.49e-13 (-)	1.14e-22 (-)
f_8	**5.75e-09** (+)	**2.98e-29** (+)	**2.97e-13** (+)	**2.12e-22** (+)	**2.41e-19** (+)	**4.40e-15** (+)
f_9	7.98e-01 (=)	**8.77e-18** (+)	**1.65e-03** (+)	**2.29e-16** (+)	2.30e-01 (=)	**1.14e-19** (+)
f_{10}	**3.66e-06** (+)	**1.02e-12** (+)	**3.57e-12** (+)	**6.45e-24** (+)	**7.07e-19** (-)	**2.73e-16** (+)
f_{11}	4.81e-03 (-)	**1.43e-12** (+)	2.17e-20 (-)	1.30e-01 (=)	3.00e-09 (-)	9.90e-29 (-)
f_{12}	1.43e-15 (-)	2.59e-01 (=)	**6.46e-18** (+)	**4.64e-06** (+)	6.51e-06 (-)	2.27e-22 (-)
f_{13}	4.20e-01 (=)	**2.09e-15** (+)	7.72e-09 (-)	**1.10e-07** (+)	5.50e-14 (-)	1.43e-21 (-)
f_{14}	8.48e-01 (=)	**7.48e-17** (+)	4.10e-09 (-)	**1.60e-07** (+)	2.32e-09 (-)	1.58e-28 (-)
f_{15}	**7.44e-10** (+)	**2.32e-22** (+)	8.89e-13 (-)	1.31e-01 (=)	3.42e-20 (-)	5.92e-15 (-)
$b/e/w*$	5/6/4	11/1/3	7/2/6	11/2/2	4/3/8	6/0/9

* LSMDE is significantly better in "b" functions, equal in "e" functions and worse in "w" functions.

TABLE VII
CONOVER'S TEST RESULTS, p-VALUES OF LSMDE VERSUS SHADE-ILS AND GL-SHADE ON THE CEC 2013 BENCHMARK FUNCTIONS FOR 1000 DIMENSIONS. SYMBOLS '+','-' AND '=' DENOTE THE LSMDE IS BETTER THAN, WORSE THAN, OR SIMILAR TO THE COMPARED ALGORITHM, RESPECTIVELY.

	Experiment 1: 1.2×10^5		Experiment 2: 6×10^5		Experiment 3: 3×10^6	
Functions	SHADE-ILS	GL-SHADE	SHADE-ILS	GL-SHADE	SHADE-ILS	GL-SHADE
f_1	7.15e-70 (-)	9.07e-36 (-)	1.34e-36 (-)	**1.34e-36** (+)	1.77e-28 (-)	**1.89e-14** (+)
f_2	**1.81e-02** (+)	2.22e-28 (-)	**7.71e-16** (+)	1.49e-28 (-)	**2.67e-19** (+)	1.75e-29 (-)
f_3	1.10e-04 (-)	1.40e-37 (-)	6.43e-03 (-)	4.06e-35 (-)	6.23e-02 (=)	9.62e-36 (-)
f_4	6.90e-01 (=)	**6.41e-16** (+)	5.73e-02 (=)	**1.82e-04** (+)	1.81e-04 (-)	7.45e-36 (-)
f_5	1.40e-03 (-)	**7.10e-28** (+)	**5.31e-04** (+)	**7.89e-22** (+)	6.74e-02 (=)	**7.91e-31** (+)
f_6	**5.36e-23** (+)	**6.88e-28** (+)	**6.84e-30** (+)	**3.40e-55** (+)	**6.67e-29** (+)	**3.68e-33** (+)
f_7	**2.69e-28** (+)	**5.86e-62** (+)	1.62e-27 (-)	**4.28e-23** (+)	2.88e-27 (-)	3.63e-39 (-)
f_8	**2.64e-33** (+)	**1.86e-64** (+)	**2.92e-27** (+)	**1.51e-38** (+)	**2.95e-32** (+)	**3.39e-27** (+)
f_9	6.64e-01 (=)	**1.43e-30** (+)	**2.90e-05** (+)	**1.37e-21** (+)	**3.97e-02** (+)	**2.72e-33** (+)
f_{10}	**3.64e-08** (+)	**1.92e-15** (+)	**3.00e-27** (+)	**3.31e-42** (+)	**2.12e-32** (+)	**2.54e-29** (+)
f_{11}	5.75e-07 (-)	**1.01e-26** (+)	1.23e-34 (-)	8.86e-03 (-)	5.87e-32 (-)	3.87e-61 (-)
f_{12}	1.07e-28 (-)	5.12e-02 (=)	**2.43e-24** (+)	**6.90e-10** (+)	3.48e-12 (-)	6.30e-35 (-)
f_{13}	1.85e-01 (=)	**1.30e-26** (+)	1.72e-30 (-)	**5.46e-27** (+)	1.07e-27 (-)	6.25e-37 (-)
f_{14}	7.56e-01 (=)	**8.69e-28** (+)	6.95e-32 (-)	**3.52e-27** (+)	1.65e-31 (-)	5.89e-60 (-)
f_{15}	**5.50e-20** (+)	**8.40e-36** (+)	3.92e-21 (-)	**2.09e-02** (+)	3.01e-34 (-)	5.62e-28 (-)
$b/e/w*$	6/4/5	11/1/3	7/1/7	12/0/3	5/2/8	6/0/9

* LSMDE is significantly better in "b" functions, equal in "e" functions and worse in "w" functions.

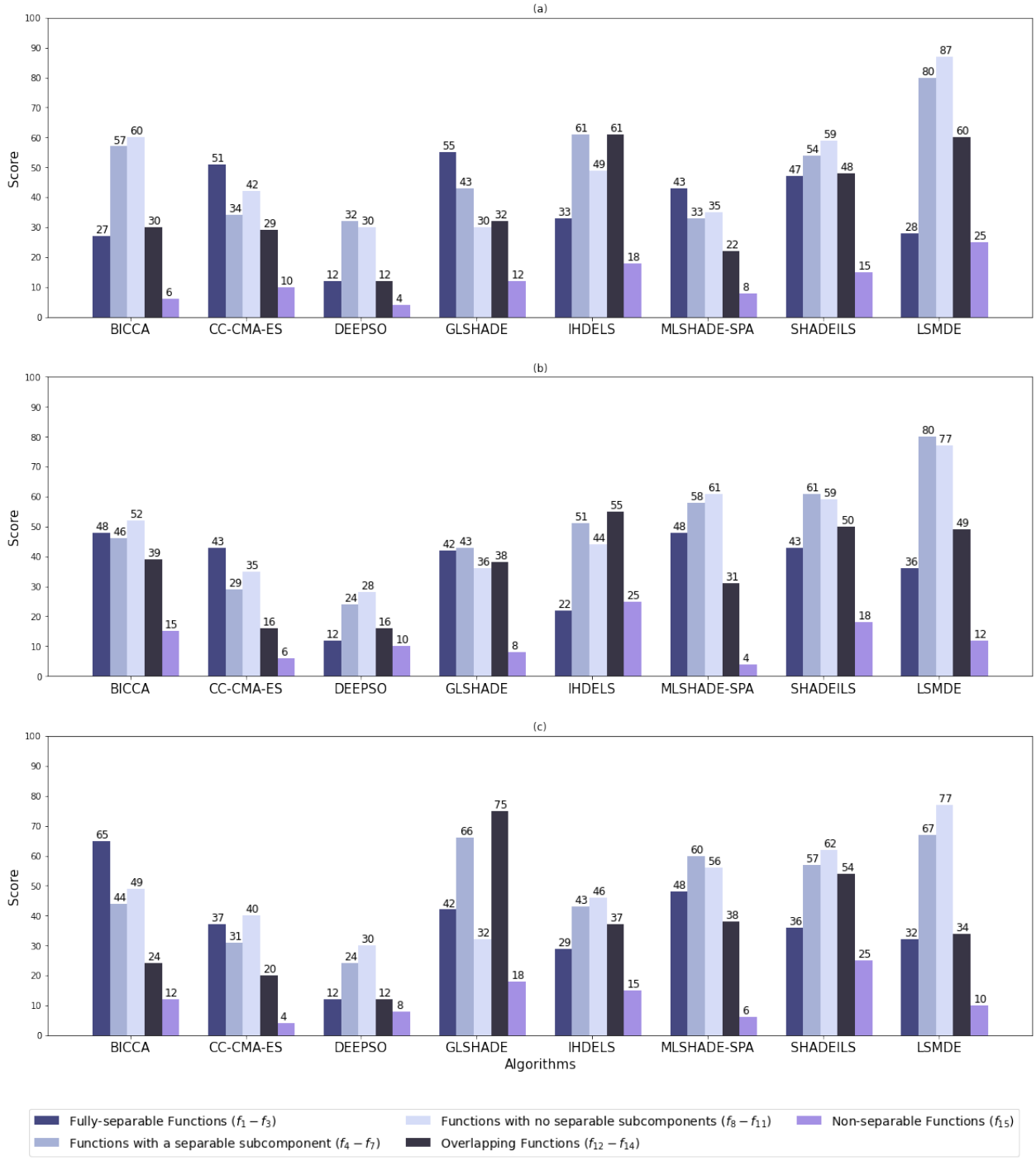

Fig. 7. Score of each competing algorithm based on the Formula One criterion for (a) 1.2×10^5, (b) 6×10^5 and (c) 3×10^6 evaluations of the objective function.

"Effect of epistasis on the performance of genetic algorithms," *Journal of Zhejiang University-SCIENCE A*, vol. 20, no. 2, pp. 109–116, 2019.

[15] Z. Ren, Y. Liang, M. Wang, Y. Yang, and A. Chen, "An eigenspace divide-and-conquer approach for large-scale optimization," *Applied Soft Computing*, vol. 99, p. 106911, 2021.

[16] X. Xue, K. Zhang, R. Li, L. Zhang, C. Yao, J. Wang, and J. Yao, "A topology-based single-pool decomposition framework for large-scale global optimization," *Applied Soft Computing*, vol. 92, p. 106295, 2020.

[17] H. G. Koçer and S. A. Uymaz, "A novel local search method for lsgo with golden ratio and dynamic search step," *Soft Computing*, vol. 25, no. 3, pp. 2115–2130, 2021.

[18] F. Schoen and L. Tigli, "Efficient large scale global optimization through clustering-based population methods," *Computers & Operations Research*, vol. 127, p. 105165, 2021.

[19] J.-R. Jian, Z.-G. Chen, Z.-H. Zhan, and J. Zhang, "Region encoding helps evolutionary computation evolve faster: A new solution encoding scheme in particle swarm for large-scale optimization," *IEEE Transactions on Evolutionary Computation*, 2021.

[20] O. Pacheco-Del-Moral and C. A. C. Coello, "A shade-based algorithm for large scale global optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2020, pp. 650–663.

[21] D. Molina, A. LaTorre, and F. Herrera, "Shade with iterative local search for large-scale global optimization," in *IEEE Congress on Evolutionary Computation*, 2018, pp. 1–8.

[22] M. Chen, W. Du, W. Song, C. Liang, and Y. Tang, "An improved weighted optimization approach for large-scale global optimization," *Complex & Intelligent Systems*, vol. 8, no. 2, pp. 1259–1280, 2022.

[23] F. Bagattini, F. Schoen, and L. Tigli, "Clustering methods for the optimization of atomic cluster structure," *The Journal of chemical*

This article has been accepted for publication in IEEE Transactions on Evolutionary Computation. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2022.3227440

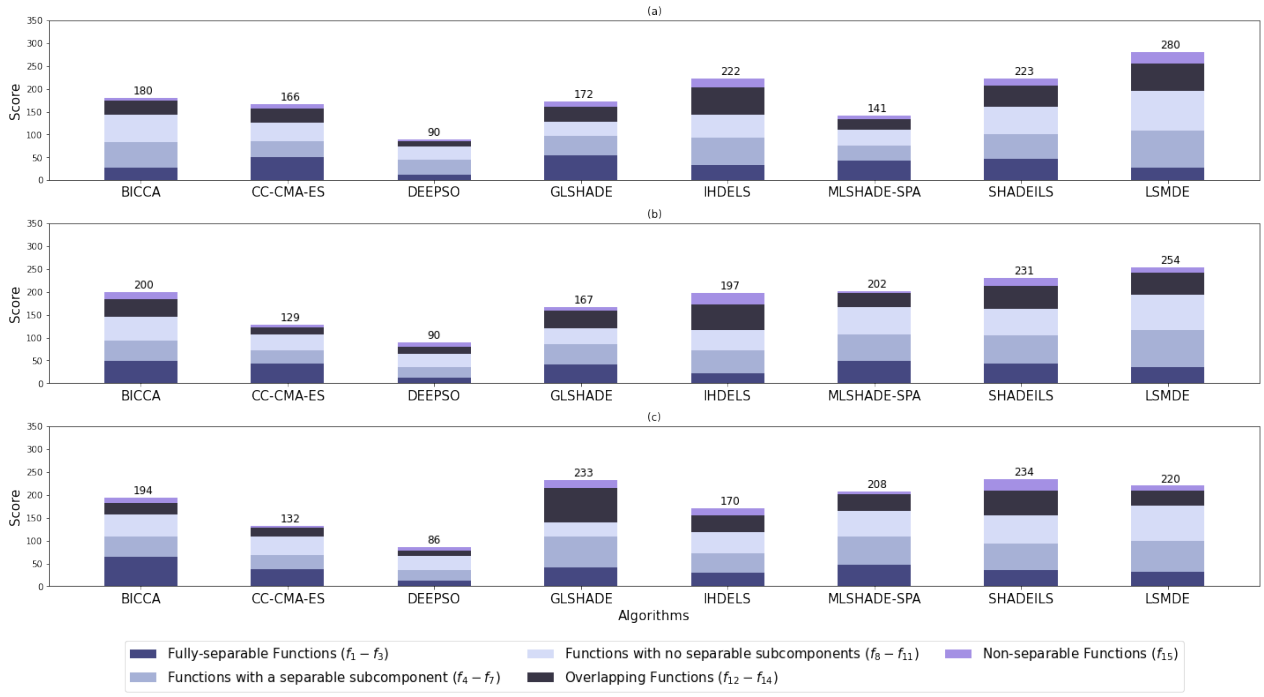

Fig. 8. General score based on the Formula One criterion for (a) 1.2×10^5 evaluations, (b) 6×10^5 evaluations and (c) 3×10^6 evaluations.

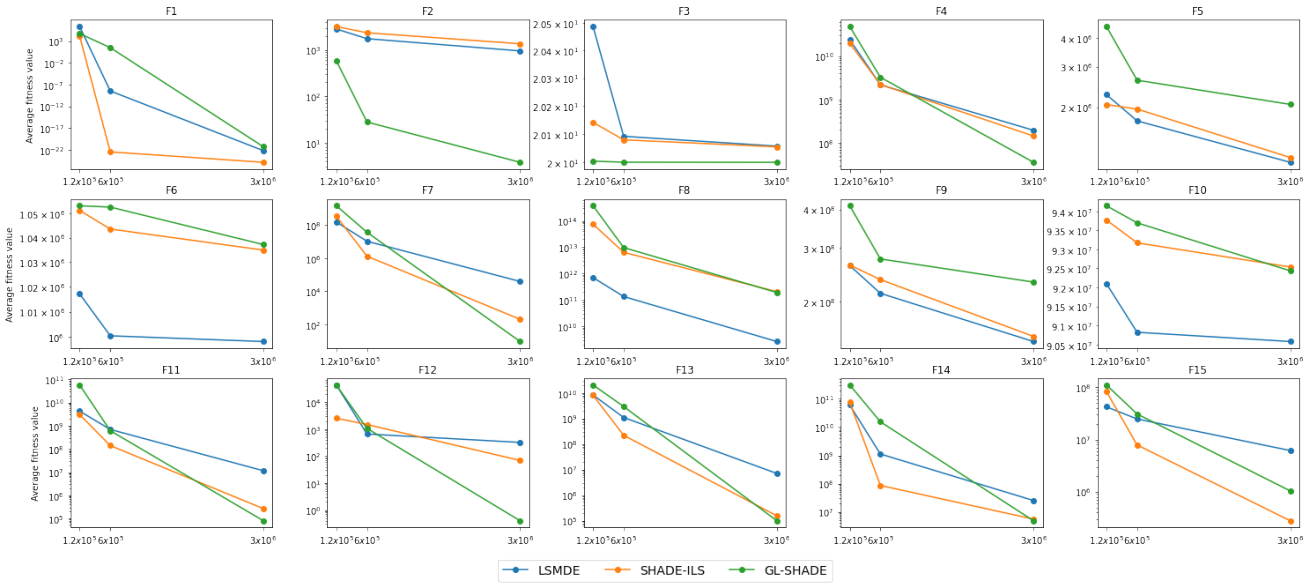

Fig. 9. Convergence functions of the three best competitor algorithms.

physics, vol. 148, no. 14, p. 144102, 2018.

[24] ——, "Clustering methods for large scale geometrical global optimization," *Optimization Methods and Software*, vol. 34, no. 5, pp. 1099–1122, 2019.

[25] 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.

[26] D. Molina and A. LaTorre, "Toolkit for the automatic comparison of optimizers: comparing large-scale global optimizers made easy," in *IEEE Congress on Evolutionary Computation*, 2018, pp. 1–8.

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

[28] J.-R. Jian, Z.-H. Zhan, and J. Zhang, "Large-scale evolutionary optimization: a survey and experimental comparative study," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 3, pp. 729–745, 2020.

[29] M. A. Potter and K. A. D. Jong, "A cooperative coevolutionary approach to function optimization," in *International conference on parallel problem solving from nature*. Springer, 1994, pp. 249–257.

[30] K. Zhang and B. Li, "Cooperative coevolution with global search for large scale global optimization," in *2012 IEEE Congress on Evolutionary Computation*. IEEE, 2012, pp. 1–7.

[31] J. Liu and K. Tang, "Scaling up covariance matrix adaptation evolution strategy using cooperative coevolution," in *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 2013, pp. 350–357.

[32] M. Yang, A. Zhou, C. Li, J. Guan, and X. Yan, "Ccfr2: A more efficient cooperative co-evolutionary framework for large-scale global optimization," *Information Sciences*, vol. 512, pp. 64–79, 2020.

[33] J. Del Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, S. Das, P. N. Suganthan, C. A. C. Coello, and F. Herrera, "Bio-inspired computation: Where we stand and what's next," *Swarm and Evolutionary Computation*, vol. 48, pp. 220–250, 2019.

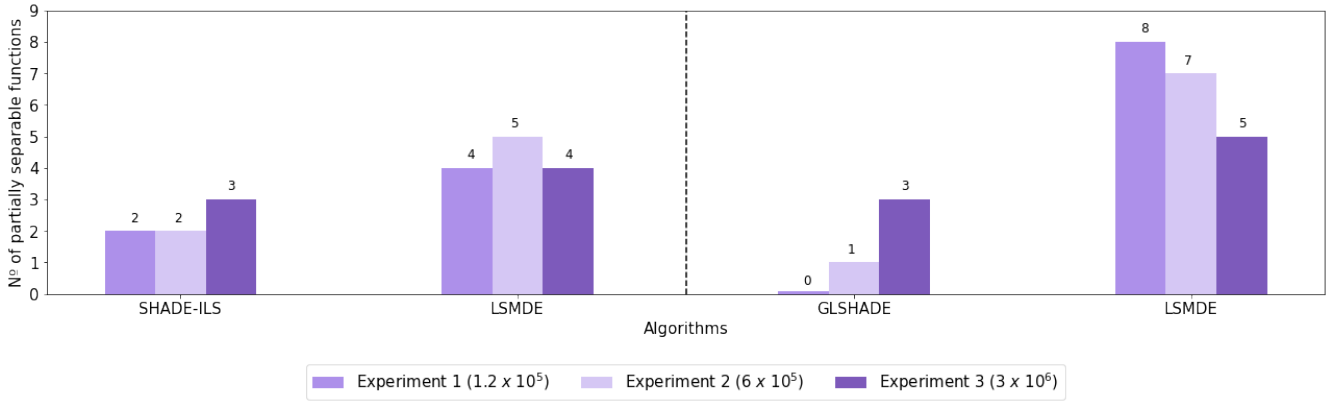

Fig. 10. Number of best partially separable functions. LSMDE vs SHADE-ILS and LSMDE vs GL-SHADE according to Conover post-hoc test for 1.2×10^5, 6×10^5 and 3×10^6 evaluations.

[34] A. LaTorre, S. Muelas, and J.-M. Pena, "Large scale global optimization: Experimental results with mos-based hybrid algorithms," in *2013 IEEE congress on evolutionary computation*. IEEE, 2013, pp. 2742–2749.

[35] C. Marcelino, P. Almeida, C. Pedreira, L. Caroalha, and E. Wanner, "Applying c-deepso to solve large scale global optimization problems," in *IEEE Congress on Evolutionary Computation*, 2018, pp. 1–6.

[36] E. Segredo, B. Paechter, C. Segura, and C. I. González-Vila, "On the comparison of initialisation strategies in differential evolution for large scale optimisation," *Optimization Letters*, vol. 12, no. 1, pp. 221–234, 2018.

[37] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, "Differential evolution mutations: Taxonomy, comparison and convergence analysis," *IEEE Access*, vol. 9, pp. 68 629–68 662, 2021.

[38] 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.

[39] A. Wagdy and A. 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, pp. 1–18, 2017.

[40] W. Deng, S. Shang, X. Cai, H. Zhao, Y. Zhou, H. Chen, and W. Deng, "Quantum differential evolution with cooperative coevolution framework and hybrid mutation strategy for large scale optimization," *Knowledge-Based Systems*, vol. 224, p. 107080, 2021.

[41] D. Molina and F. Herrera, "Iterative hybridization of de with local search for the cec'2015 special session on large scale global optimization," in *IEEE Congress on Evolutionary Computation*, 2015, pp. 1974–1978.

[42] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *2013 IEEE congress on evolutionary computation*. IEEE, 2013, pp. 71–78.

[43] 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.

[44] L.-Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 2008, pp. 3052–3059.

[45] J. L. Morales and J. Nocedal, "Remark on "algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound constrained optimization"," *ACM Transactions on Mathematical Software (TOMS)*, vol. 38, no. 1, pp. 1–4, 2011.

[46] H. Ge, M. Zhao, Y. Hou, Z. Kai, L. Sun, G. Tan, Q. Zhang, and C. P. Chen, "Bi-space interactive cooperative coevolutionary algorithm for large scale black-box optimization," *Applied Soft Computing*, vol. 97, p. 106798, 2020.

[47] W. Dong, Y. Wang, and M. Zhou, "A latent space-based estimation of distribution algorithm for large-scale global optimization," *Soft Computing*, vol. 23, no. 13, pp. 4593–4615, 2019.

[48] S. Mahdavi, S. Rahnamayan, and K. Deb, "Opposition based learning: A literature review," *Swarm and evolutionary computation*, vol. 39, pp. 1–23, 2018.

[49] B. Kazimipour, X. Li, and A. K. Qin, "Effects of population initialization on differential evolution for large scale optimization," in *IEEE Congress on Evolutionary Computation*, 2014, pp. 2404–2411.

[50] S. Mahdavi, S. Rahnamayan, and K. Deb, "Partial opposition-based learning using current best candidate solution," in *IEEE Symposium Series on Computational Intelligence*, 2016, pp. 1–7.

[51] N. Kishore Kumar and J. Schneider, "Literature survey on low rank approximation of matrices," *Linear and Multilinear Algebra*, vol. 65, no. 11, pp. 2212–2244, 2017.

[52] J. E. Tougas and R. J. Spiteri, "Updating the partial singular value decomposition in latent semantic indexing," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 174–183, 2007.

[53] X. Wang and X. Jin, "Understanding and enhancing the folding-in method in latent semantic indexing," in *International Conference on Database and Expert Systems Applications*. Springer, 2006, pp. 104–113.

[54] S. Dasgupta and A. Gupta, "An elementary proof of the johnson-lindenstrauss lemma," *International Computer Science Institute, Technical Report*, vol. 22, no. 1, pp. 1–5, 1999.

[55] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.

[56] Y. Song, D. Wu, A. Wagdy Mohamed, X. Zhou, B. Zhang, and W. Deng, "Enhanced success history adaptive de for parameter optimization of photovoltaic models," *Complexity*, vol. 2021, 2021.

[57] S. Chakraborty, A. K. Saha, S. Sharma, S. K. Sahoo, and G. Pal, "Comparative performance analysis of differential evolution variants on engineering design problems," *Journal of Bionic Engineering*, pp. 1–21, 2022.

[58] D. M. Blei and M. I. Jordan, "Variational inference for dirichlet process mixtures," *Bayesian analysis*, vol. 1, no. 1, pp. 121–143, 2006.

[59] C. Rasmussen, "The infinite gaussian mixture model," *Advances in neural information processing systems*, vol. 12, 1999.

[60] M. N. Omidvar, X. Li, and K. Tang, "Designing benchmark problems for large-scale continuous optimization," *Information Sciences*, vol. 316, pp. 419–436, 2015.

[61] Z. Michalewicz, "Quo vadis, evolutionary computation?" in *IEEE World Congress on Computational Intelligence*. Springer, 2012, pp. 98–121.

[62] Z. Michalewicz, M. Schmidt, M. Michalewicz, and C. Chiriac, *Adaptive business intelligence*. Springer, 2006.

[63] J. Sobieszczanski-Sobieski and R. T. Haftka, "Multidisciplinary aerospace design optimization: survey of recent developments," *Structural optimization*, vol. 14, no. 1, pp. 1–23, 1997.

[64] J. T. Allison, M. Kokkolaras, and P. Y. Papalambros, "Optimal partitioning and coordination decisions in decomposition-based design optimization," *Journal of Mechanical Design*, 2009.

[65] D. Molina, A. R. Nesterenko, and A. LaTorre, "Comparing large-scale global optimization competition winners in a real-world problem," in *IEEE Congress on Evolutionary Computation*, 2019, pp. 359–365.

[66] A. Soria-Alcaraz Jorge, C. Martín, P. Héctor, and M. A. Sotelo-Figueroa, "Comparison of metaheuristic algorithms with a methodology of design for the evaluation of hard constraints over the course timetabling problem," in *Recent Advances on Hybrid Intelligent Systems*. Springer, 2013, pp. 289–302.

[67] E. Ostertagova, O. Ostertag, and J. Kováč, "Methodology and application of the kruskal-wallis test," in *Applied Mechanics and Materials*, vol. 611. Trans Tech Publ, 2014, pp. 115–120.

[68] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.