Evaluating Center-Seeking and Initialization Bias: The case of Particle Swarm and Gravitational Search Algorithms (Extended Version)

Mohsen Davarynejad*†, Jan van den Berg* and Jafar Rezaei*
*Faculty of Technology, Policy and Management, Delft University of Technology, Jaffalaan 5, Delft 2628BX, The Netherlands
†Department of Radiation Oncology, Erasmus Medical Center—Daniel den Hoed Cancer Center, Groene Hilledijk 301, Rotterdam 3075EA, The Netherlands

Email: {M.Davarynejad; J.vandenBerg; J.Rezaei}@tudelft.nl

Abstract—Complex optimization problems that cannot be solved using exhaustive search require efficient search metaheuristics to find optimal solutions. In practice, metaheuristics suffer from various types of search bias, the understanding of which is of crucial importance, as it directly pertinent to the problem of making the best possible selection of solvers. Two metrics are introduced in this study: one for measuring center-seeking bias (CSB) and one for initialization region bias (IRB). The former is based on “center offset”, an alternative to “center offset”, which is a common but, as will be shown, inadequate approach to analyzing the center-seeking behavior of algorithms. The latter is proposed on the grounds of “region scaling”. The introduced metrics are used to evaluate the bias of three algorithms while running on a test bed of optimization problems having their optimal solution at, or near, the center of the search space. The most prominent finding of this paper is that some of them suffer from a specific search bias [12], [36]; they tend to perform best when the optimum is located at or near the center of the search space. Consequently, when comparing the quality of the solutions found by a set of metaheuristics on a series of benchmark problems with optimal solution near the center of the search space, the comparison becomes unfair.

To remedy this unfairness, the so-called center offset (CO) [37] approach was proposed which changes the borders of the search space in such a way that the optimal solution is no longer located in the center of the search space. Basically, the CO approach changes the search space of the original problem by reducing it on one side and expanding it at the other. When comparing a set of algorithms qualitatively, the comparison is valid since interference tends to be reduced when all the contenders are submitted to the same set of benchmarks, no matter if the shifting has introduced some degree of increase/decrease in the complexity of the search. Our goal, here, is to supplement the comparison by developing a trade-off between exploration and exploitation. This is an important notion when it comes to allocating scarce resources to the exploration of new possibilities and the exploitation of old certainties.

Nature has been the original inspiration for many types of metaheuristics. Two distinct classes of nature-inspired optimization algorithms are evolutionary algorithms (EA), and swarm intelligence (SI)-based algorithms. Some popular members of the former class are genetic algorithm (GA) [25] and differential evolution (DE) [46], [57]. Successful instances of swarm intelligence-based algorithms are particle swarm optimization (PSO) [28] and the gravitational search algorithm (GSA) [47].

Arguments for allowing bias in the search methods state that a learning system may be shaped towards the important features of a problem [60]. So when more information about the solution becomes available then the user may take advantage of specific types of search bias. However general purpose optimizers make no assumption on the problem at stake. Studying the properties of these algorithms, it turns out that some of them suffer from a specific search bias [12], [36]; they tend to perform best when the optimum is located at or near the center of the search space. Consequently, when comparing the quality of the solutions found by a set of metaheuristics on a series of benchmark problems with optimal solution near the center of the search space, the comparison becomes unfair.

I. INTRODUCTION

Consider a search scenario in a finite continuous search space \( E \subset X \) defined by

\[
E = \bigotimes_{d=1}^{D} [L_d^d, U_d^d],
\]

with the objective of locating \( x^* \in E \), where \( f(x^*) \) is the extremum of a function \( f(x) : E \rightarrow \mathbb{R} \), and where \( L_d^d \) and \( U_d^d \) are respectively the lower and upper bound of the search domain at dimension \( d \). Optimization problems are to be found in such diverse arenas as engineering, business, medicine, etc. [5]. Without loss of generality, a minimization problem is considered here.

In contrast to exhaustive search which looks into every entry in the search space, metaheuristics [23] are strategies which guide the search process iteratively, in many cases by making important notion when it comes to allocating scarce resources to the exploration of new possibilities and the exploitation of old certainties.
quantitative measures that can assist the observer in evaluation of the “degree” of CSB of a certain search algorithm. Quantitative measures are succinct and are the preferred disclosure form, not only for a) a comparison of the degree of CSB in a set of search algorithms, but also when the task is b) to examine if a single search algorithm has any CSB at all.

On the basis of these observations, we decided to examine generic methods for evaluating the search bias of different algorithms. In this paper, we limit ourselves to two metrics; one for measuring center-seeking bias, and one for initialization bias. These metrics are used to evaluate the behavioral bias of several algorithms related to swarm optimization and gravitational search.

Section II-A presents the center offset (CO) approach (introduced in [1]) to capture the center-seeking behavior of population based algorithms. Then proceeds by arguing that CO is not a reliable test. Then in Section II-B a solution for the unknown change in problem complexity as a result of using CO approach when changing the center of the search space is reported and a metric to measure the CS behavior of population based optimization techniques is presented. A metric to measure initialization region bias is then presented in Section III. In Section IV-A and IV-B PSO and GSA are briefly summarized. Next, in Section IV-C, the search bias of the GSA in favor of center of the search space is conceptualized. Then we present a solution to partially dilute the search bias of GSA. In Section V We have analyzed the search behavior of the studied contenders by setting up a number of tests similar to those proposed by Angeline [1] on a set of several widely used numerical benchmark problems with various optimization characteristics. In addition to the studied benchmark problems, a gene regulatory network model identification problem is studied to assess the performance of the GSA and the presented modification of GSA. Section VI presents discussions and provides a framework that enables a fair comparison of optimization heuristics. The last section highlights conclusions and provides suggestions for future research.

II. A METRIC FOR MEASURING CENTER-SEEKING BIAS

A. Understanding the assumptions underlying center offset

According to the No Free Lunch theorem [62], all learning systems will expose equal performance over all possible cost functions. This implies that, in order to efficiently solve an optimization problem, they should be tailored to the salient problem-specific characteristics. Where there is no available information on the problem at hand, as with various real-world applications, some search biases known to us are not often of service. Such biases include center-seeking (CS) behavior and initialization region bias (IRB), the foci of this study.

When comparing nature-inspired metaheuristic algorithms, a symmetric search space can be misleading when the optimal solution is located at, or near, the center of the search space. In such a case, one must account for CS behavior in order to draw valid conclusions from an experiment [8]. One attempt to deal with CS bias is called center offset (CO). This is a common approach to negating the centrist bias of an optimization algorithm [3]. The underlying assumption of CO is that the complexity of a problem does not change as a result of moving the optimal solution from the center of the search space; this is an assumption that is discussed in greater detail below.

When applying CO, the optimization problem $f(x)$ is changed to $f(x - C)$ where C is the location of the new center. CO is equivalent to expanding the search space from one side, for each dimension $d$, and to shrinking it on the other side, without changing the distance $\|U_x^d - L_x^d\|$ between the lower bound $L_x^d$ and the upper bound $U_x^d$. When the objective of a test is to measure the search bias of an algorithm, CO is of not of any use. This is because a change in the complexity of a problem is not explicitly controlled: without any additional information, the complexity of the problem might increased, decreased, or even remained the same. As a consequence, any observed difference in the performance of an algorithm cannot, to any degree of certainty, be associated with the CS bias of the algorithm; it may also have been caused by an (unknown) change in the problem complexity.

Figure 1 shows an example of an increase in problem complexity (due to an increase in the number of local optimal solutions) as a result of shifting the search window when the objective is to locate the minimum of the following function:

$$f(x) = 10(x - 0.2)^2 + \sin(\pi x), 0.1 \leq x \leq 0.3.$$  \hspace{1cm} (2)

In this case, due to an increased problem complexity, the average performance of any metaheuristic is expected to deteriorate whether or not the algorithm possesses CS bias. Consequently no hypothesis can be made on the CS behavior of an optimization algorithm.

Assuming we know that the problem complexity decreases, some decision making around CS behavior becomes possible. If a certain algorithm shows a better performance, one can...
conclude that this algorithm has no observable CS bias, since
we would otherwise have observed a deterioration in its
performance, i.e., a deterioration in the best found fitness
during optimization. Figure 2 summarizes the concerns that
need to be addressed before drawing any conclusion regarding
the outcome of the standard CO test, which provides an
argument in favor of searching for an alternative CO-approach,
in which it is known that the problem complexity always
decreases. In this study, the ξ-CO approach is introduced to
remove the uncertainties on change in problem complexity
(dashed arrows in Figure 2).

In ξ-CO, the search space is downsized asymmetrically, as
a result of which the problem complexity always decreases and
the algorithm is expected to locate a near optimal solution
more quickly and with greater precision if there is no center-
seeking bias. This makes it possible to test the hypotheses
on CS behavior of an optimization algorithm considering a
benchmark problem with (i) a symmetric search space and
(ii) the optimal solution near the center of the search space.
Let us assume that \( L^d_x = -U^d_x \) and that \( U^d_x > 0 \), as is the case
for most of the problems studied here. In ξ-CO, the search
space is downsized asymmetrically by modifying the lower
bound of the search space \( L^d_x \) according to

\[
L^d_x = L^d_x + \frac{\xi}{100} \times \frac{|U^d_x - L^d_x|}{2},
\]

where \( \xi \in [\xi_L, \xi_U] \) is the percentage of downsizing the search
space and where \( \xi_L \) and \( \xi_U \) are the predefined lower and upper
bound of \( \xi \), respectively.

Observe that CO is only worthwhile for optimization prob-
lems whose support extends outside of the initial boundary
constraints. With the proposed ξ-CO approach this shortcom-
ing does not arise.

B. A metric for center-seeking bias

After identifying ξ-CO as an appropriate approach for
analyzing CS behavior of optimization heuristics, there is still
a need to quantify the observations on the CS bias behavior,
i.e., a metric is needed. By executing a series of runs when
gradually increasing, with a predefined step size of \( \xi_s \), the
percentage \( \xi \) of downsizing the search space from a lower limit
\( \xi_L \) to an upper limit \( \xi_U \), one can measure the best-fitneseach
optimization algorithm can attain. Because randomly chosen
initializations affect the outcome, experiments under equal
conditions are usually repeated several times, say \( r_f \) time,
yielding a data of the form \( (\xi, f^x_\xi) \) ∈ \( \mathbb{R}^2 \) when \( r \in [1, r_f] \).
Based on these observations, an estimation of best-of-run \( f^x_\xi \)
as function of \( \xi \)

\[
f^x_\xi = \text{CSB}_{\xi_L, \xi_U}^\xi \xi + c_1
\]

(4)
can be calculated, where \( \text{CSB}_{\xi_L, \xi_U}^\xi \) is the slope of the
regression line. The slope \( \text{CSB}_{\xi_L, \xi_U}^\xi \) has been selected as
the metric to analyze CS behavior of optimization heuristics.
For minimization problems, in case \( \text{CSB}_{\xi_L, \xi_U}^\xi \geq 0 \), the best
fitness found increases, implying the presence of CS bias
behavior (because the quality of the solutions found does
not improve when complexity is reduced). Similarly, in case
\( \text{CSB}_{\xi_L, \xi_U}^\xi < 0 \), the best fitness found decreases, implying that
there is no observable CS bias behavior.

In special cases, \( \xi \) may be changed from its lower limit
to its upper limit without intermediate steps. In this case the
the corresponding metric is referred to as \( \text{CSB}^\xi_{\xi_L, \xi_U} \). Comparing
\( \text{CSB}^\xi_{\xi_L, \xi_U} \) and \( \text{CSB}^\xi_{\xi_L, \xi_U} \), the latter has a greater
generalization ability and is the preferred metric for comparing
algorithms under study.

Note, finally, that the proposed metric is not restricted to
situations where the search space is downsized according to
ξ-CO. The original CO approach may still be used, namely,
in cases where the problem at hand is well-known and the
problem complexity due to center offset remains unchanged.
Under such circumstances, the metric \( \text{CSB}^\xi_{\xi_L, \xi_U} \) can be used
as well. In that case, \( \xi \in [\xi_L, \xi_U] \) is the percentage by which the
center of the search space is offset.

III. A METRIC FOR INITIALIZATION REGION BIAS

Most of today’s real-world optimization problems are for-
mulated in environments that undergo continual change, re-
ferred to as dynamic optimization problems [2]. When the
global optimal solution changes, the population members have
to move along an extended path with many local optima.
To test the sensitivity of PSO on the search initialization, as
well as its ability to move from the initial search space to
more promising regions, Angeline [1] proposed reducing the

![Fig. 2: CO at a glance. CO is a common approach used to test the center-seeking behavior of heuristic optimization algorithms. The results of the test may not be interpretable. The proposed ξ-CO results are interpretable, because the problem complexity is always reduced.](image)
initialization region, referred to as Region Scaling (RS) [37]. This initialization is adopted here as well, where the algorithm is initialized deliberately in a portion of the search space. A notable example of a class of algorithms suffering from sufficiently generating offsprings outside a given initial population, specially when the size of the population is small relative to the search space, is GA with Unimodal Normal Distribution Crossover (UNDX) [41].

Region Scaling (RS) [37] is an approach to quantitatively observe if a search algorithms has any IRB. By shrinking the initialization region (IR), an algorithm with no IRB will perform not worse than when the IR covers the entire de-

The observations have a linear relationship with the dependent variable $y$ and the independent variable $x$ can be calculated directly from the observed results.

$$\hat{f}(\zeta) = \text{IRB}_{\zeta_\text{L} - \zeta_\text{U}} \cdot \zeta + e_2$$  (5)

where $\text{IRB}_{\zeta_\text{L} - \zeta_\text{U}}$ is the slope of the regression line fitted to $(\zeta, f(\zeta)) \in \mathbb{R}^2$.

In special cases where $\zeta_\text{L}$ is equal to $\|\zeta_\text{U} - \zeta_\text{L}\|$ the metric is referred to as $\text{IRB}_{\zeta_\text{L} - \zeta_\text{U}}$. To measure the initialization region bias, the $\text{IRB}_{\zeta_\text{L} - \zeta_\text{U}}$ has a greater generalization ability compared to $\text{IRB}_{\zeta_\text{L} - \zeta_\text{U}}$.

Observe that while search algorithms may perform better when they are initialized in the whole search space and benefit from knowing the search space, one with lower IRB is preferable over one with higher IRB.

IV. THREE POPULATION-BASED METAHEURISTICS

The primary goal of this study is to assay CSB and IRB of a set of widely used and well-established metaheuristics. We do not aim at giving an exhaustive experimental comparison on a wide range of alternative search algorithms, rather we focus on a set of well benchmark instances. This section respectively formulates the particle swarm algorithm as proposed in [6] and gravitational search algorithm [47] in addition to presenting a modification of GSA.

A. A Brief Tour of the particle swarm optimization

Swarm intelligence, an emerging collective behavior of interacting agents with examples of ant colony [19] and bee colony [26], is a popular source of inspiration for the design of optimization algorithms. Particle swarm optimization (PSO) [28] is a successful instance of a nature-inspired algorithm for solving global optimization problems. A number of advantages have been attributed to PSO, making it a choice candidate as a benchmark algorithm. The standard PSO algorithm is suited to handle nonlinear nonconvex optimization problems with fast convergence characteristics. In this study, PSO is a reasonable choice for comparison as it does not have any bias towards the center of the search space [27].

In classical PSO, every particle is a solution moving in a $D$-dimensional search space. A collection of particles is known as swarm. Each particle $i$ has a position, $x_i \in \mathbb{R}^D$, a velocity, $v_i \in \mathbb{R}^D$ and the best position found so far, $p_i \in \mathbb{R}^D$.

PSO uses two independent random variables, $r_1, r_2 \sim \mathcal{U}(0,1)$, scaled by constants $C_1$ and $C_2$. The constants $C_1$ and $C_2$ are known as learning rates and they influence the maximum step size a particle can take in a single iteration, representing the confidence of a particle on its best performance and that of the global best respectively. The movement equations of every particle $i$ in $1, 2, \ldots , D$, are given by expressions (6) and (7).

$$v_i^d = wv_i^d + C_1r_1^d (p_i^d - x_i^d) + C_2r_2^d (g^d - x_i^d), \quad (6)$$

$$x_i = x_i + v_i, \quad (7)$$

where $w$ is a predefined constant representing the confidence of particle on its own movements and $p_i^d$ and $g^d$ are personal best and global best positions respectively. $S$ is the number of particles in the swarm.

To ensure convergence by avoiding explosion, Clerc et al. [6] introduces the constriction factor and modifies the velocity update equation as follows:

$$v_i^d = \chi (v_i^d + C_1r_1^d (p_i^d - x_i^d) + C_2r_2^d (g^d - x_i^d)), \quad (8)$$

where $\chi = \frac{2}{|2-\varphi - \sqrt{\varphi^2 - 4\varphi}|}$ and $\varphi = C_1 + C_2$, $\varphi > 4$.

B. A Brief Tour of the GS Algorithm

Gravitational search algorithm (GSA) [47] is a relatively new technique that has been empirically shown to perform well on many optimization problems [4], [17], [20], [24], [32], [33], [42], [45], [48], [55]. GSA was inspired by the Newton’s law of universal gravitation. In its original version, GSA scatters particles in a feasible region of the search space, where they interact with each other under Newton’s gravitational force and move in the search area seeking an optimal design variable. GSA shares features with several competing schemes, for instance by sharing information between solutions. In contrast to EAs where solutions die at the end of each generation, in PSO and GSA, solutions survive throughout the optimization process, providing a substantial source of information for the population when searching the global optimum.
In GSA, like in many other population based optimization techniques, to guide the population in the search space $E$, some measure of discrimination is needed, referred here as a fitness of each candidate solution $x_i$. Each candidate solution is a particle with a mass $M_i$. A good solution is analogous to a particle with a high mass, while a poor solution represents a particle with a low mass. A particle with a high mass resists change more than one with a low mass and tends to have higher impact on other particles, thereby sharing its features with low quality solutions. The attractive gravitational force governs the movement of the particles in the search space. The search begins by an attractive force with a strength and direction as a function of the mass of particle itself, the mass of other particles and its relative distance to the other particles. The force is applied to static particles of one under which their position in the next time step changes and they gain velocity. The quantity of the resulting force is determined by Newton’s gravitational law. A solution with a higher mass exerts a stronger force compared with a smaller mass. The kinetic energy stored in particles is a form of memory, allowing them to steer their movement under the influence of their memory and external forces. The sum of the force field $F_i$ and the particle’s kinetic energy, induced from its velocity and mass, is the total force acting on them, which together with its current position $x_i(t)$, determines the particles next position $x_i(t + 1)$ in the search space.

In original GSA [47], the mass of particles, considering its quality, is assigned as follows:

$$M_i = \frac{m_i}{\sum_{j=1}^{S} m_j}, i = 1, 2, \ldots, S \quad (9)$$

where

$$m_i = \frac{f(x_i) - \max_{j \in \{1, \ldots, S\}} f(x_j)}{\min_{j \in \{1, \ldots, S\}} f(x_j) - \max_{j \in \{1, \ldots, S\}} f(x_j)} \quad (10)$$

and $S$ is the number of particles. The resulting gravitational force acting on particle $i$ in direction $d$ is determined using Equation (11).

$$F_i^d = \sum_{j \in K_{best}} r_{ij} F^d_{ij} \quad (11)$$

where $K_{best}$ is a set of particles with the highest mass, $r_{ij} \sim \mathcal{U}(0,1)$ and $F^d_{ij}$ is the gravitational force exerted by particle $j$ on particle $i$. To provide a better exploration in the early iterations $|K_{best}|$ is set at $S$ in the beginning; however the exploration must be decreased gradually. Therefore choosing a decremented function for $|K_{best}|$ increases the exploitation of the algorithm when the number of iterations increases.

The force exerted by particle $j$ acting on particle $i$ is defined as:

$$F^d_{ij} = G \frac{M_i \times M_j}{R_{ij} + \varepsilon} (x^d_i - x^d_j) \quad (12)$$

where $R_{ij}$ is Euclidian distance between particles $i$ and $j$, and $G$, the gravitational constant initialized at $G_0$ is determined using Equation (13) as:

$$G = G_0 e^{-\alpha \frac{t}{MaxIteration}} \quad (13)$$

where $\alpha$ is algorithmic tuning parameter and $MaxIteration$ is the maximum iteration.

The equations of motion of every particle are described using (14) and (15) as:

$$v_i(t + 1) = R \times v_i(t) + \frac{F_i}{M_i} \Delta t, \quad (14)$$

$$x_i(t + 1) = x_i(t) + v_i(t + 1) \cdot \Delta t, \quad (15)$$

where $\Delta t = 1$, $R \sim \mathcal{U}(0,1)$ is an array of size $D$ corresponding to each element in vector $v_i$.

C. mdGSA, a mass-dispersed GSA

In GSA, an increase in the number of particles changes the mass assigned to them as a result of an increase in the denominator of the Equation (9). This increase in the denominator smooths out the difference between the mass of particles, making them in absolute terms, more equal in exerting an attractive force and equally resistant to change in their position as a result of the applied gravitational force. The swarm can be seen as one particle with a uniform mass distribution. Under the Newtonian gravitational force, this brings the particles closer to the center of the swarm, resulting in an increase in the density of swarm. As a result, they move more quickly towards the center of the search space [12]. This may explain the center-seeking behavior of standard GSA. It is against this backdrop that a different GSA called mass-dispersed gravitational search algorithm (mdGSA) is devised and tested here.

A more intense discrimination of solutions is proposed in Simulated Big Bounce (SBB) algorithm [15]. SBB is a global search algorithm that is inspired by the Big Bounce theory (a cosmological oscillatory model of the Universe), that, next to exploitation, applies robust exploration in order to escape local minima. In this approach, based on their fitness, the particles are assigned a mass in the range of $|L_M, U_M|$, $g$, the function that maps the fitness to the mass $g : \mathbb{R} \rightarrow \mathbb{R}$, $f(x_i) \mapsto g(f(x_i))$, $\forall x_i \in E$ can be any monotonically nondecreasing (and possibly time varying) function in principle with real values defined on a the set of fitness of particle $x_i$ whose value is non-negative for $f(x_i)$. We take $g$ as a linear time-invariant strictly increasing function as follows [15]:

$$M_i = g(f(x_i)) = L_M + (U_M - L_M) \frac{f(x_i) - \max_{j \in \{1, \ldots, S\}} f(x_j)}{\min_{j \in \{1, \ldots, S\}} f(x_j) - \max_{j \in \{1, \ldots, S\}} f(x_j)} \quad (16)$$

Algorithm 1 captures the framework of mdGSA’s basic steps.
Algorithm 1 Pseudo code of mass-dispersed gravitational search algorithm (mdGSA)

Input: Search space $E$, fitness function $f$, $S$, $G_0$, $\alpha$
1: Initialize particle’s Location, $x = (x_1, \ldots, x_S)^T$
2: while $t < MaxIteration$ do
3:    Fitness calculation
4:    Update $M_i$, $\forall i = 1, \ldots, S$ $\triangleright$ According to (16)
5:    Update $G$ $\triangleright$ According to (13)
6:    Update attractive force $F^d_i$, $\forall i = 1, \ldots, S$
7:    Update $x_i$, $\forall i = 1, \ldots, S$ $\triangleright$ According to (14)
8:    $t + +$ $\triangleright t$ is the number of iterations
9: end while
Output: $x^*$ and $f(x^*)$

V. EXPERIMENTAL RESULTS

When comparing the effectiveness of different optimization heuristics, a standard performance measure is the best fitness a certain algorithm can reach within a predefined number of function evaluations. This is based on the assumption that the dominating factor in measuring computational effort is fitness evaluation, which is usually valid for complex optimization problems [14], [16], [52]. In the experiments, this, is modeled as if the maximum computational resource budget available to carry out a task were limited, which is equivalent to a situation where the maximum time budget for which the best solution has to be delivered is limited.

In general, although the studied optimization algorithms can be simply extended and adapted for real-world optimization problem, such adaptation may require more elaborate mechanisms. One example of this is constraint-handling. It is well-known that in real-world optimization problems there are normally constraints of different types (e.g., related to the geometry of structural elements to completion times, etc.) that must be satisfied for a solution to be acceptable. Traditionally, penalty functions have been used to handle constraints [7]. However, because of the several problems associated to penalty functions (e.g., the definition of appropriate penalty values is normally a difficult task that has a serious impact on the performance of the optimizer), some authors have proposed alternative constraint-handling approaches that require less critical parameters and perform well across a variety of problems (see for example [7], [35], [51]).

In the experiments described in this section, the common parameters used in each algorithm, such as population size and total number of fitness evaluation, where chosen to be the same. Unless indicated otherwise, the population size is set at 50 and the maximum number of fitness evaluation, MaxIteration, is set at 100,000. For Gene regulatory network (GRN) model identification problem (Section V-B), the maximum number of fitness evaluation is set at 200,000.

1) PSO settings: The PSO parameters across the experiments have been $\phi = 4.1$, $\phi_1 = \phi_2$ and $\chi = 0.729$, which is equivalent to setting $C_1 = C_2 = 1.496$ and $\omega = 0.729$ [6].
2) GSA settings: The GSA parameters are as follows: $G_0$ is set at 100, $\alpha$ is set at 20, $K_{best}$ is set at number of particles, $S$, and is linearly decreased to 1 in the final iteration, (MaxIteration) [47].
3) mdGSA settings: The common setting are GSA settings. The upper and lower bound of mass are set at 1 and 0.01, respectively.

Because the optimization techniques under study are stochastic in nature, for a given function of a given dimension 30 independent runs where executed for all $\xi$ and $\zeta$ values. Throughout the experiments discussed in this study, the population size and maximum fitness evaluation remain fixed, although it is well known that these control-parameters affect the performance of the algorithms. The reason not to change the parameters was primarily the motivation of our study in exposing the center-seeking behavior and IRB of GSA, rather than emphasizing its performance under different control-parameter settings. The second reason reales to the assumption that end-users do not know much about the algorithmic parameters for their optimization problem.

A. Experiment 1: Standard optimization problems

From the test beds studied in [47], [63], those with varying dimensions are used in this study to capture the CS behavior of GSA [47] in addition to those studied in [34]. The test beds along with their characteristics are listed in Table I.

Because the primary objective of this study is to specify the center-seeking behavior of GSA, the Schwefel function is excluded, since its optimal solution is not close to the center of the search space. In Table I, $D$ is the dimension of function. The optimal solution $f(x^*)$ for all the adopted test functions is located at $[0, \ldots, 0]_D$ with the exception of Dixon-Price, that has its optimal solution located at $2 \cdot \frac{4}{D}$ for $d = 1, 2, \ldots, D$ as well as Levy and Rosenbrock with optimal solution at $[1, \ldots, 1]_D$. Of the 14 adopted test studies, half are unimodal, while the others are multimodal. The set contains five separable and nine non-separable functions. A separable function can be decomposed into $D$ one-dimensional functions.

The performance of the algorithms is evaluated from both accuracy and robustness perspectives. Accuracy is the degree of precision of an optimization algorithm in locating an optimal solution. An algorithm with a higher accuracy tends to come closer to the optimal solution. Accuracy is studied in two different settings for optimization problems, $\xi$-Accuracy and $\zeta$-Accuracy. $\xi$-Accuracy refers to the performance of the optimization algorithms (OAs) under study when the center of the search space changes, while $\zeta$-Accuracy refers to the

1Although constraint-handling techniques are very important in real-world optimization problems, their discussion is beyond the scope of this article, due to space limitations. Interested readers are referred to other references for more information on this topic (see for example [35], [50]).

2Note that, in [47], the Rosenbrock function (also known as Banana problem) is treated as a unimodal test function when $D$ is set at 30, while it is indeed multimodal when the problem dimensions is more than three [18], [54].
performance of OAs when the initialization region changes. Robustness is defined here as the degree of bias of an optimization heuristics on the center of the search space or the initialization region. A robust optimization algorithm has no CS or IR bias.

In our experiments, the metrics are measures in a log-linear scale, because the best-of-run found by each algorithm, in many cases, changes several orders of magnitude as a result of $\xi$-CO and/or $\xi$-RS.

1) $\xi$-CO test results: Figures 3 and 5 are the test results of the $\xi$-CO on the studied standard benchmark problems when $D$ is set at 50 and 100, respectively. The x-axis is $\xi$ and the y-axis is the performance of each OA, averaged over $r_f = 30$ independent runs. Throughout this study $\xi_L$ and $\xi_U$ are set at 5 and 45 respectively and the step size $\xi_s$ is set at 5. These choices for $\xi_L$ and $\xi_U$ are based on the assumption that an optimal solution of a real-world problem is usually neither at the center of the search space, nor at the boundaries, suggesting $\xi = 0$ and $\xi = 50$ are not interesting cases to study.

Figures 3 and 5 show that, as a result of downsizing the search space, the performance of the PSO algorithm is nearly a straight line in most of the experiments. The performance of the GSA deteriorates quickly by moving the optimal solution from the center of the search space. mdGSA falls somewhere in between.

Tables II and IV summarize the $CSB_{\xi_a}^{\xi_b}$ when $D$ is set at 50 and 100, respectively. In each table, asterisk symbols are used to denote no statistically significant association between the observed change in estimation of best-of-run as a result of change in $\xi$ using F-statistics. Statistical testing is performed to determine whether or not $CSB_{\xi_a}^{\xi_b}$ measures are zero, in addition to testing if $CSB_{\xi_a}^{\xi_b}$ (mdGSA) is statistically smaller than that of $CSB_{\xi_a}^{\xi_b}$ (GSA).

In the case of Step function, $F_{1,4}$, the optimal solution, 0, is attainable under a relatively wide range of design values. This, in log-linear scale, leaves us with no way to fit a line to the observed performance. Hence, the Step function is excluded from Tables II and IV. As a replacement for it, the convergence curve of some selected $\xi$ values are visualized in Figure 7.

For each of the nine $\xi$ values on each of the 14 problems, the optimization methods are statistically compared using pairwise contrast. The number of times an OA has a statistically significant superiority (SSS) compared to other optimization algorithms on a total of $9 \times 14$ problems is shown in Tables III and V. We also report the number of times an optimization approach achieves the best result, best mean and best median when it is statistically superior to others. As an example, the number of times an algorithm performs best is the number of times a) it is statistically superior to others and b) it has the best fitness over 30 runs compared to other competing algorithms. In addition, the number of times the worst result is achieved is reported.

a) Results on 50D problems: First, we evaluate the robustness of each algorithm when the dimension of the optimization problems is set at 50. For each optimization algorithm, $CSB_{\xi_a}^{\xi_b}$ are reported in Table II. The slope of fitted line describing center-seeking bias of PSO (Md = -0.3019) was not significantly different from zero (Wilcoxon signed-rank, $W=33$, $p=0.2071$) while for GSA (Md = 5.8795, Wilcoxon signed-rank, $W=1$, $p=2.44E-4$) and mdGSA (Md = 0.9621, Wilcoxon signed-rank, $W=8$, $p=0.0030$) the fitted line had a slope significantly different from zero. Interestingly, the observed $CSB_{\xi_a}^{\xi_b}$ (GSA) was significantly higher than $CSB_{\xi_a}^{\xi_b}$ (mdGSA) (Wilcoxon rank sum test, $W=224$, $p=0.069$).

Out of total of 9*14 experiments each repeated 30 times, when $D = 50$, mdGSA and GSA are competing closely when looking at the number of times they were statistically superior to the others (Table III). As a result, a statistical test of significance was performed on median of fitness they both can achieve under different settings of studied optimization problems. For that, logarithmic transformation of the median of fitness values was performed in the first place (because the Wilcoxon test assumes that the distribution of the data, although not normal, is symmetric). Wilcoxon paired sample test ($W=3399$, $p=0.8865$) confirms that there is no significant difference between the performance of GSA and mdGSA. Note that, due to logarithmic transformation of the medians, the Step function is excluded from the test, which means that the test is performed on 13 test problems, each with 9 different $\xi$ values.

So, while mdGSA and GSA come in joint first place, PSO, with only two cases of SSS, comes second (Table III). While PSO is the most robust of the algorithms under study, it did not perform better than the others in terms of its $\xi$-accuracy.

The picture changes when we look at other measures, for instance the worst solutions over the 30 runs. While GSA and mdGSA come close in terms of their statistical superiority, mdGSA shows the worst fitness in only 13 cases, compared to 47 cases for GSA, which suggests that GSA is more susceptible to trapping around a local optimum and missing the global optimum.

As mentioned before, Figures 3 and 5 are presenting the

| Table II: CSB_{\xi_a}^{\xi_b} of the studied algorithms when D is set at 50. |
|-------------------|-------------------|-------------------|
| $CSB_{\xi_a}^{\xi_b}$ (GSA) | $CSB_{\xi_a}^{\xi_b}$ (mdGSA) | $CSB_{\xi_a}^{\xi_b}$ (PSO) |
| Ackley | 9.219 | 1.25 | -0.49* |
| Dixon | 1.217 | 0.01371 | 0.4631 |
| Griewank | 5.879 | 2.195 | 0.8024* |
| Levy | 29 | 12.42 | 4.599 |
| Penalty1 | 31.3 | 5.48 | -1.96* |
| Penalty2 | 49.75 | 0.856* | -2.843 |
| Quaic | -0.05537* | 0.01948* | -1.119 |
| Rastrigin | 2.02 | 1.503 | 0.2341 |
| Rosenbrock | 0.3789 | 0.9621 | -0.04733* |
| Schwefel222 | 4.5 | 1.074 | -0.3019 |
| Schwefel12 | 11.01 | -0.5769 | 0.1159* |
| Schwefel121 | 11.79 | -0.02214* | -0.3442 |
| Sphere | 4.285 | 0.02609* | -0.6474 |
Fig. 3: Mean of the best fitness obtained under the $\xi$-CO test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 50$. a) Ackley, b) Dixon-Price, c) Griewank, d) Levy, e) Penalized1, f) Penalized2, g) Quartic, h) Rastrigin.
Fig. 3: (continued) Mean of the best fitness obtained under the $\xi$-CO test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 50$. i) Rosenbrock, j) Schwefel P2.22 k) Schwefel P1.2, l) Schwefel P2.21, m) Sphere, n) Step.
average of the performance of each of the contenders when $\xi$ changes from 5 to 45 with step size of 5. In addition to that, Figures 4 and 6 are providing the performance of each of the contenders for various $D$ at 100 considering CSB when $\xi$ is set at 45. These observations are consistent with our previous findings.

With regard to the case where the dimension of the search space is set at 100, the results of PSO basically remain the same (Figure 7.b). While, in the beginning of the search, for $\xi = 5$, GSA has the greatest reduction of fitness among its competitors, it has the highest performance deterioration when the center of the search space is moved, confirming its strong center-seeking bias. mdGSA, although defeating GSA under equal settings in all three cases, shows a degrading performance when $\xi$ is set at 45. These observations are consistent with our previous findings.

2) $\xi$-RS test results: Figures 8 to 9 contain the test results of the $\xi$-RS on the test problems when $D$ is 50 and 100, respectively. The x-axis is the percentage of shrinking the centered search space, and the y-axis is the number of times the best median of fitness values is obtained.

When $D = 50$ (Figure 7.a) as a result of moving the center of the search space, the performance of both PSO and mdGSA does not change very much. For GSA, there is a clear deterioration in performance when the center of the search space is moved. When $\xi = 5$ (which basically means that the optimal solution is near to the center of the search space), GSA locates the optimal solution very quickly and defeats its competitors, while it fails to locate the optimal solution when it is removed from the center of the search space ($\xi = \{25, 45\}$).

When $D = 100$, GSA has the greatest reduction of fitness among its competitors, it has the highest performance deterioration when the center of the search space is moved, confirming its strong center-seeking bias. mdGSA, although defeating GSA under equal settings in all three cases, shows a degrading performance when $\xi$ is set at 45. These observations are consistent with our previous findings.

### Table III: Comparison results of the three studied algorithms (GSA, mdGSA and PSO) when $\xi$ changes from 5 to 45 and when $D$ is set at 50.

<table>
<thead>
<tr>
<th>Simulation results</th>
<th>GSA</th>
<th>mdGSA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSS $^a$</td>
<td>57</td>
<td>51</td>
<td>2</td>
</tr>
<tr>
<td>Best $^b$</td>
<td>56</td>
<td>33</td>
<td>1</td>
</tr>
<tr>
<td>Worst $^c$</td>
<td>48</td>
<td>12</td>
<td>63</td>
</tr>
<tr>
<td>Best mean $^d$</td>
<td>43</td>
<td>49</td>
<td>2</td>
</tr>
<tr>
<td>Best median $^e$</td>
<td>57</td>
<td>51</td>
<td>2</td>
</tr>
</tbody>
</table>

*a # of times the fitness values are statistically superior.
*b # of times the best fitness value is obtained.
*c # of times the worst fitness value is obtained.
*d # of times the best mean of fitness values is obtained.
*e # of times the best median of fitness values is obtained.

### Table IV: CSB$_{5-45}^\frac{\xi}{\xi}$ of the studied algorithms when D is set at 100.

<table>
<thead>
<tr>
<th>Simulation results</th>
<th>CSB$_{5}^{5-45}$ (GSA)</th>
<th>CSB$_{45}^{5-45}$ (mdGSA)</th>
<th>CSB$_{5}^{5-45}$ (PSO)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ackley</td>
<td>26.59</td>
<td>15.35</td>
<td>-0.1096*</td>
</tr>
<tr>
<td>Dixon</td>
<td>5.409</td>
<td>2.789</td>
<td>-0.0233*</td>
</tr>
<tr>
<td>Griewank</td>
<td>4.043</td>
<td>2.863*</td>
<td>-0.6713</td>
</tr>
<tr>
<td>Levy</td>
<td>10.98</td>
<td>7.655</td>
<td>-0.3052</td>
</tr>
<tr>
<td>Penalty1</td>
<td>7.644</td>
<td>5.933</td>
<td>-0.1683</td>
</tr>
<tr>
<td>Penalty2</td>
<td>12.43</td>
<td>5.624</td>
<td>-0.5562</td>
</tr>
<tr>
<td>Quartic</td>
<td>44.37</td>
<td>5.285</td>
<td>-1.158</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>1.943</td>
<td>1.627</td>
<td>0.2927</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>12.34</td>
<td>2.723</td>
<td>0.06779*</td>
</tr>
<tr>
<td>Schwefel222</td>
<td>17.93</td>
<td>12.84</td>
<td>-0.1035*</td>
</tr>
<tr>
<td>Schwefel122</td>
<td>14.21</td>
<td>3.023</td>
<td>0.897</td>
</tr>
<tr>
<td>Schwefel121</td>
<td>1.737</td>
<td>3.475</td>
<td>2.457</td>
</tr>
<tr>
<td>Sphere</td>
<td>54.84</td>
<td>2.904</td>
<td>-0.5122</td>
</tr>
</tbody>
</table>

### Table V: Comparison results of the three studied algorithms (GSA, mdGSA and PSO) when $\xi$ changes from 5 to 45 and when $D$ is set at 100.

<table>
<thead>
<tr>
<th>Simulation results</th>
<th>GSA</th>
<th>mdGSA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSS $^a$</td>
<td>20</td>
<td>67</td>
<td>11</td>
</tr>
<tr>
<td>Best $^b$</td>
<td>19</td>
<td>58</td>
<td>5</td>
</tr>
<tr>
<td>Worst $^c$</td>
<td>71</td>
<td>0</td>
<td>55</td>
</tr>
<tr>
<td>Best mean $^d$</td>
<td>14</td>
<td>63</td>
<td>11</td>
</tr>
<tr>
<td>Best median $^e$</td>
<td>20</td>
<td>67</td>
<td>11</td>
</tr>
</tbody>
</table>

*a # of times the fitness values are statistically superior.
*b # of times the best fitness value is obtained.
*c # of times the worst fitness value is obtained.
*d # of times the best mean of fitness values is obtained.
*e # of times the best median of fitness values is obtained.
Fig. 4: Test results of the $\xi$-CO for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 50$. a) Ackley, b) Dixon-Price, c) Griewank, d) Levy, e) Penalized1, f) Penalized2, g) Quartic, h) Rastrigin.
Fig. 4: (continued) Test results of the $\xi$-CO for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 50$. i) Rosenbrock, j) Schwefel P2.22 k) Schwefel P1.2, l) Schwefel P2.21, m) Sphere, n) Step.

initialization region and the y-axis is the performance averaged over 30 independent runs. An algorithm with no IRB has the opportunity to explore areas outside the initialization region.

Throughout this study, $\zeta_L$ is set at 10 when $\zeta_L = 0$ and $\zeta_U = 90$. A line best fitted to $10^L*30$ observations has a slope of $\text{IRB}_{\zeta}$.

Tables VI and VIII present an estimation of the degree of change in the quality of best-of-run of each optimization heuristic as a result of shrinking the initialization region, when the dimension of problems are set at 50 and 100, respectively. Here, again, in each table asterisks symbols are used to denote no statistically significant association between observed change in estimation of best-of-run as a result of change in $\zeta$ using F-statistics.

For the same reason as $\xi$-CO, the Step function is excluded from the Tables VI and VIII. Instead, for some selected $\zeta$ values, its convergence curve is visualized in Figure 10. To compare $\zeta$-accuracy of the optimization heuristics under study, we look at the number of times one is statistically superior to the others and the number of times one has the
Fig. 5: Mean of the best fitness obtained under the \( \xi \)-CO test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when \( D = 100 \). a) Ackley, b) Dixon-Price, c) Griewank, d) Levy, e) Penalized1, f) Penalized2, g) Quartic, h) Rastrigin.
Fig. 5: (continued) Mean of the best fitness obtained under the $\xi$-CO test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 100$. i) Rosenbrock, j) Schwefel P2.22 k) Schwefel P1.2, l) Schwefel P2.21, m) Sphere, n) Step.
Fig. 6: Test results of the $\xi$-CO for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 100$. a) Ackley, b) Dixon-Price, c) Griewank, d) Levy, e) Penalized1, f) Penalized2, g) Quartic, h) Rastrigin.
worst fitness (Tables VII and IX). For each optimization heuristic, the number of test problems equals $10^4 \times 14$ (10 $\zeta$ values for each of the 14 problems). We also report the number of times an optimization method has the best performance, best mean and best median when it is statistically superior. The number of times each has the worst fitness is reported as well.

a) Results on 50D problems: In 10 of the 13 optimization problems, the performance of the PSO algorithm degrades as a result of shrinking the initialization space. However compared to GSA, its IRB$^{10-90}$ is small. The performance of the GSA on eight optimization problem degrades under $\zeta$-RS test. The mdGSA is more robust to the initialization region and is nearly a straight line in 11 of the 13 experiments. The slope is significantly different than zero in only two cases. In both of these cases, the slope is negative, meaning that the performance of the algorithm increases as a result of shrinking the initialization region. A possible explanation for this will be suggested later.

For GSA (Wilcoxon signed-rank, $W=227.5$, $p=0.0012$)
and PSO (Wilcoxon signed-rank, \( W=240.5 \), \( p=1.60\times10^{-4} \)), the IRB\_10\_90 values are significantly different than zero, while in the case of mdGSA (Wilcoxon signed-rank, \( W=162.5 \), \( p=0.1655 \)), the IRB\_10\_90’s does not differ significantly from zero, which suggests that both GSA and PSO are not robust with regard to initialization region.

In the combination of 14 test problems and 10 initializations regions, GSA and mdGSA are competing closely when looking at the number of times each of them is significantly superior to the others (in 75 and 51 cases, respectively). So, similar to \( \xi\)-CO, a statistical test of significance is performed on median of fitness each of them can obtain on different settings of studied optimization problems when a logarithmic transformation of the median of fitness values is performed. Wilcoxon signed-rank test\(^3\) results (\( W=3981 \), \( p=0.5205 \)) confirm

\[
\begin{array}{llll}
\text{TABLE VI: IRB\_10\_90. Initialization region bias when } \xi \text{ changes from 0 to 90 and when } D \text{ is set at 50.} \\
\hline
& \text{IRB\_10\_90 (GSA)} & \text{IRB\_10\_90 (mdGSA)} & \text{IRB\_10\_90 (PSO)} \\
\hline
\text{Ackley} & 1.162 & -0.005047* & 3.556 \\
\text{Dixon} & 0.001543* & -8.645e-015* & 0.1817 \\
\text{Griewank} & 3.462 & -0.48* & 0.1506* \\
\text{Levy} & 1.255* & -0.5644* & 3.842 \\
\text{Penalty1} & 13.02 & 0.6169* & 0.2494* \\
\text{Penalty2} & 22.8 & -0.6841* & 0.5677* \\
\text{Quartic} & -0.02836* & -0.01201* & 0.8033 \\
\text{Rastrigin} & -0.000824* & -0.03876 & 0.1463 \\
\text{Rosenbrock} & 0.7701 & 0.01091* & 0.2249 \\
\text{Schwefel222} & -0.007248* & -0.002421* & 0.4908 \\
\text{Schwefel12} & 6.375 & 0.01072* & 0.8677 \\
\text{Schwefel121} & 5.495 & -0.04105 & 0.3853 \\
\text{Sphere} & 30.28 & 0.003999* & 0.2297 \\
\hline
\end{array}
\]

\(^3\)Note that, again, the Step function is excluded from the test due to logarithmic transformation of the medians, which means that the test is performed on 13 test problems, each with 10 different \( \xi \) values.
TABLE VII: Comparison results of the three studied algorithms (GSA, mdGSA and PSO) when \( \zeta \) changes from 0 to 90 and when D is set at 50.

<table>
<thead>
<tr>
<th>Simulation results</th>
<th>GSA</th>
<th>mdGSA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSS (^a)</td>
<td>75</td>
<td>51</td>
<td>0</td>
</tr>
<tr>
<td>Best (^b)</td>
<td>74</td>
<td>43</td>
<td>0</td>
</tr>
<tr>
<td>Worst (^c)</td>
<td>58</td>
<td>11</td>
<td>69</td>
</tr>
<tr>
<td>Best mean (^d)</td>
<td>56</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>Best median (^e)</td>
<td>75</td>
<td>51</td>
<td>0</td>
</tr>
</tbody>
</table>

\(^a\) # of times the fitness values are statistically superior.

\(^b\) # of times the best fitness value is obtained.

\(^c\) # of times the worst fitness value is obtained.

\(^d\) # of times the best mean of fitness values is obtained.

\(^e\) # of times the best median of fitness values is obtained.

Table VII summarizes the results.

b) Results on 100D problems: In Table VIII, the slope of the line fitted to observed best-of-run fitness values for each benchmark problem is presented when D is set at 100. IRB\(_{10}^{0\rightarrow90}\) (PSO) has significantly positive associations with \( \zeta \) in all cases. GSA has a significantly positive IRB\(_{10}^{0\rightarrow90}\) in 11 of the total of 13 cases, while IRB\(_{10}^{0\rightarrow90}\) (mdGSA) has no significant positive associations with \( \zeta \). This suggests that GSA (Wilcoxon signed-rank, W=247, p=9.97E-5) and mdGSA (Wilcoxon signed-rank, W=156, p=.29) both have significant initialization region bias when D is set at 100, while mdGSA (Wilcoxon signed-rank, W=156, p=.29) has IRB values that are significantly close to zero.

As was the case when the dimension of the test problems is set at 50, mdGSA again has some negative IRB values. An intuitive explanation for this observation goes as follows. Due to high mass discrimination of mdGSA, when the initialization region is small compared to the entire design space, the particle with the highest fitness that is equivalent to highest mass is better able to direct the entire swarm towards the optimal solution. As a result the algorithm performs slightly better compared to when the initialization is taken in the entire search space. It is important to point out that the improvement is not visible in most of the cases from Figures 8 and 9 and that this improvements, in all 13 cases, has no significant correlation with \( \zeta \), confirmed by analysis of covariance (F-test). When compared to GSA, mdGSA has significantly less IRB (Wilcoxon rank sum test, W=246, p = 1.65E-4). So, in terms of robustness in change in IR, mdGSA defeats its competitors.

Summarized in Table IX, mdGSA presents higher \( \zeta \)-accuracy compared to its two competitors. With 87 cases of significant superiority out of total of 14*10 cases, mdGSA ranked in the first place, followed by GSA, with 26 cases, and PSO, with zero cases. Wilcoxon paired-sample test (W=1735, p=4.5869E-9) also confirms the superiority of mdGSA when looking at the logarithmic transformation of the median of their performance.

GSA has the worst fitness in 75 cases, and PSO in 65 cases. mdGSA has the smallest number of worst fitness compared to that of GSA and PSO, confirming its superiority over its competitors in terms of \( \zeta \)-accuracy. This again suggests that mdGSA is less susceptible to premature convergence to local optimum when compared to PSO and GSA.

c) \( \zeta \)-RS test results on Step function: Again for the Step function, the results are not presented in Tables VI and VIII. So the performance of the studied algorithms when \( \zeta = \{10, 50, 90\} \) are presented in Figure 10.

When \( D = 50 \) (Figure 10.a) GSA\(_{10}\) has a sharp fitness decrease. The performance degrades significantly as a result of shrinking the initialization region (GSA\(_{50}\) and GSA\(_{90}\)). For mdGSA, the performance slightly changes as a result of change in IR, but the pattern was not clear. The performance of PSO was basically the same under different IRs.
Fig. 8: Mean of the best fitness obtained under the $\zeta$-RS test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 50$. a) Ackley, b) Dixon-Price, c) Griewank, d) Levy, e) Penalized1, f) Penalized2, g) Quartic, h) Rastrigin.
Fig. 8: (continued) Mean of the best fitness obtained under the $\zeta$-RS test condition for GSA (blue), mdGSA (red) and PSO (black) when $D = 50$. a) Dixon-Price, b) Quartic, c) Schwefel P1.2, d) Schwefel P2.21, e) Sphere, f) Step.

GSA$_{10}$, when the dimension is set at 100 (Figure 10.b), has a sharp fitness change and becomes almost stagnant after approximately 300 iterations on average. As a result of shrinking the search space, there is a clear pattern in deterioration of the performance GSA. PSO exhibits similar patterns as expected, shrinking the initialization region does not appear to affect its performance. For mdGSA, on the Step function and when the dimension of the problem is set at 100, moving the optimum away from the center of the search space improves the performance slightly. These observations are compatible with our former findings.

B. Experiment 2: Gene regulatory network model identification

Gene regulatory network (GRN) model identification can be a good real-world application to test the center-seeking behavior and convergence speed of the optimization algorithms, since the optimal solution is, naturally, not at the center of the search space. Moreover the problem is highly nonlinear and complex [21], [29], [56]. A short introduction to GRN is
Fig. 9: Mean of the best fitness obtained under the $\zeta$-RS test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 100$. a) Ackley, b) Dixon-Price, c) Griewank, d) Levy, e) Penalized1, f) Penalized2, g) Quartic, h) Rastrigin.
Fig. 9: (continued) Mean of the best fitness obtained under the ζ-RS test condition for GSA (blue square), mdGSA (red triangle) and PSO (black asterisk) when $D = 100$. i) Rosenbrock, j) Schwefel P2.22 k) Schwefel P1.2, l) Schwefel P2.21, m) Sphere, n) Step.
Fig. 10: Performance comparison on Step function for $\zeta = \{10, 50, 100\}$ when a) $D = 50$, b) $D = 100$.

1) Gene regulatory Network: The activation and inhibition of genes are governed by factors within a cellular environment and outside of the cell. This level of activation and inhibition of genes is integrated by gene regulatory networks (GRNs), forming an organizational level in the cell with complex dynamics [9]. GRNs in a cell are complex dynamic network of interactions between the products of genes (mRNAs) and the proteins they produce, some of which in return act as regulators of the expression of other genes (or even their own gene) in the production of mRNA. While low cost methods to monitor gene expression through microarrays exist, we still know little about the complex interactions of these cellular components. Mathematical modeling of GRNs is becoming popular in the post-genome era [30], [31]. It provides a powerful tool, not only for a better understanding of such complex systems, but also for developing new hypotheses on underlying mechanisms of gene regulation. The availability of high-throughput technologies provides time course expression data, and a GRN model built by reverse engineering, may explain the data [39]. Since many diseases are the result of polygenic and pleiotropic effects controlled by multiple genes, genome-wide interaction analysis is preferable to single-locus studies. Readers looking for more information on GRN might refer to Schlitt [53].

2) S-system gene network model: Usually, sets of ordinary differential equations (ODEs) are used as mathematical models for these systems [59]. S-system approaches, on the other hand, use time-independent variables to model these processes. Assuming the concentration of $N$ proteins, mRNAs, or small molecules at time $t$ is given by $y_1^t, y_2^t, \ldots, y_i^t, \ldots, y_N^t$, S-systems model the temporal evolution of the $i$th component at time $t$ by power-law functions of the form (17).
The observed expression pattern is the relative mean quadratic discrepancy between the model and the system. To guide the search space, some measure of discrimination is needed. The most commonly used quality assessment criterion is the relative mean quadratic discrepancy between the observed expression pattern \( y^t_i \) and the model output \(  \hat{y}^t_i \) [40].

$$ f = \sum_{i=1}^{N} \sum_{t=1}^{T} \left( \frac{\hat{y}^t_i - y^t_i}{y^t_i} \right)^2, \quad (18) $$

where \( T \) represents the number of time points.

To assess the performance of the methodologies studied here, a gene regulatory network consist of two genes generated by the parameters provided in Table X is adopted [58]. In the original implementation, the search space for \( \alpha_i \) and \( \beta_i \) is limited to \([0.0, 20]\) and for \( g_{ij} \) and \( h_{ij} \) to \([-4.0, 4.0]\) and \( g_{ij}^0 \) and \( h_{ij}^0 \) are set at 0.7 and 0.3, respectively. The gene expression levels are plotted in Figure 11 and each consist of 50 time course of expression level per gene. To study the effect of initialization region on the converge of the optimization algorithms the initialization set to cover part of the search space. In this study, \( \alpha_i \) and \( \beta_i \) is initialized in \([10, 20]\) and both \( g_{ij} \) and \( h_{ij} \) to \([2.0, 4.0]\).

4) Results: The fitness transitions of studied methodologies are plotted in Figure 12. All algorithms discussed here start with a randomly generated population of solutions, which means they all start with close fitness values. The Figures are averaged over 30 independent runs.

All three OAs start with a sharp fitness decrease in the beginning. GSA almost stagnates after approximately 2,000 iterations. mdGSA shows a much better progression compared to PSO and GSA.

As shown in Table XI, the results of the proposed mdGSA are better than those of GSA when the standard cut-off for considering a \( p \)-value for a statistically significant difference is set at \( p < 0.05 \). While mdGSA is not significantly superior to GSA, it shows a better performance, with a smaller standard deviation.

<table>
<thead>
<tr>
<th>Network parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha_i )</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

Fig. 11: Target time dynamics of adopted gene network.

Fig. 12: Performance comparison of the GSA, mdGSA and PSO on GRN parameter identification.

VI. DISCUSSIONS

The heuristics we studied are compared on the basis of their robustness and accuracy (performance). We divided the evaluation of robustness into the following two assessments:
...When studying center-seeking bias, PSO is found to be the most appropriate optimization algorithm (OA). It shows no observable CS bias. mdGSA comes second, followed by GSA. Statistical comparisons confirm that mdGSA holds less bias towards the center of the search space when compared to GSA.

- When studying initialization region bias, the performance of the PSO deteriorates (statistically) when the IR is tightened. This is consistent with existing literature [43] and confirms that an efficient swarm initialization improves performance. GSA also showed significant deterioration in its performance. mdGSA showed no statistical change in its performance as a result of shrinking the IR.

From the change in performance that comes from a change in the center of the search space, as well as a change in IR, we can conclude that mdGSA has less CS bias and less IRB compared to GSA. It is thus more robust.

From an exploration-exploitation perspective, ζ-RS provides us with better understanding of the behavior of the optimization algorithms. Algorithms with high IRB have limited abilities to explore promising regions outside of the IR. This is associated with the algorithm’s weak exploration. Looking at the CS bias metric, GSA holds a strong search bias towards the center of the search space. It even does not have sufficient exploration to search beyond initialization region, which lack is confirmed by the metric proposed to measure IRB. This puts into question the robustness of GSA. mdGSA, on the other hand, while it enjoys less center-seeking bias, has enough exploration which is confirmed by its statistically zero IRB. This is thanks to its dispersed mass assignment procedure. Consequently, mdGSA has a high level of robustness.

As had been shown, mdGSA has a number of negative IRBs. This observation is in line with the statement made in [44]. There we see that the initial population is beneficial when it guides the population towards the global optimum, and that, whenever possible, the alleviation of the negative effects of this bias should be sought. Among the optimization techniques which we studied, mdGSA is the only one that takes advantage of the initialization region to guide the population.

The evaluation of accuracy is divided into the following two assessments:

- In low-dimensional optimization problems, both GSA and mdGSA outperform PSO. There is no significant difference between GSA and mdGSA when counting the number of times one is statistically superior to the others. This is confirmed by a statistical test. However, when we collect total number of worst solutions, mdGSA performs better than GSA.

- In high-dimensional optimization problems, mdGSA performs better than both PSO and GSA when we consider the number of times one is statistically superior to the others. The same results is achieved when total of worsts solutions are looked at.

Table XII presents a summary of the comparison of the optimization heuristics examined in this study. It does so in terms of both their robustness and this accuracy. Robustness is compared by looking at the metrics presented to measure CSB and IRB. Accuracy is compared by looking at the quality of solutions found for benchmark optimization problems under two different settings, ζ-accuracy and ζ-accuracy. To summarize, in terms of robustness when the center of the search space is changed, PSO is the best of those we studied. In terms of robustness when changes are made in the initialization region, mdGSA places first. When looking at ζ-accuracy and ζ-accuracy, GSA and mdGSA come joint first for low-dimensional problems, while mdGSA places first for high-dimensional problems. mdGSA comes in first place when looking at the number of times it has the worst fitness compared to the other contenders. High numbers of resulting in worst fitness suggests the susceptibility of PSO and GSA to becoming trapped in local optima.

Note that it is not in our interest to suggest not to use GSA because of its strong search bias. User should be aware, rather, of the way in which it might affect their needs. It is also notable that, in this work, the setting are those recommenced in original work. It must be noted, however, that changing the algorithmic parameter settings and stopping criteria, the benchmark functions, and even the grading criteria may change the results and conclusions. In spite of these caveats, we believe these preliminary results are a promising indication of the success of the proposed mdGSA on a wide range of optimization problems.

VII. CONCLUSIONS AND FUTURE WORK

Metaheuristics are a family of approximate methods used to find good solutions to computationally difficult optimization problems. While some optimization heuristics suffer from various types of search bias, a review of the literature reveals a lack of an appropriate quantification metric. The major contribution of this study is the development of metrics that measure the center-seeking and initialization region bias of.
optimization heuristics. We also propose an alternative for center offset, as we identified its assumption does not always hold.

Using the proposed metrics, the center-seeking (CS) bias and initialization region bias (IRB) of GSA are exposed. Our interest in this study was not to improve the performance of GSA, which can be archived by the integration of useful heuristics. Rather, it was about presenting a solution to dilute its CS behavior and its IRB. Inspired by our recently introduced global optimization process, we established a “mass-dispersed” version of GSA called mdGSA. PSO served as our benchmark because it shows no bias towards the center of the search space [27].

To further substantiate the limitations and capabilities of GSA and mdGSA in dealing with real-world optimization problems, we want to apply them to a wider range of problems, such as structural design optimization [11], [16], the detection of hidden information in a spread-spectrum watermarked signal [10], and problems of traffic control [13].

Several optimization heuristics have evolved in the last decade to facilitate solving optimization problems (see for example [22], [38], [49]), some of which suffer from different types of search bias [61]. The framework presented in this study appears to be a viable approach when it comes to comparing different optimization heuristics. As part of our future work, we are interested in using the framework proposed here to contrast different optimization heuristics suitable to handling high-dimensional and complex real-world optimization problems.

REFERENCES


<table>
<thead>
<tr>
<th>Function name</th>
<th>Mathematical Representation</th>
<th>Characteristic</th>
<th>Original Search space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ackley</td>
<td>$f_1(x) = -20 \exp \left( -0.2 \sqrt{\sum_{i=1}^{D} x_i^2} \right) - \exp \left( \frac{1}{D} \sum_{i=1}^{D} \cos (2\pi x_i) \right) + 20 + e$</td>
<td>MN</td>
<td>$[-30, 30]^D$</td>
</tr>
<tr>
<td>Dixon-Price</td>
<td>$f_2(x) = (x_1 - 1)^2 + \sum_{i=2}^{D} \left( x_i^2 - x_{i-1} \right)^2$</td>
<td>UN</td>
<td>$[-10, 10]^D$</td>
</tr>
<tr>
<td>Griewank</td>
<td>$f_3(x) = \frac{1}{\prod_{i=1}^{D} x_i^2} - \prod_{i=1}^{D} \cos \left( \frac{\pi x_i}{D} \right) + 1$</td>
<td>MN</td>
<td>$[-600, 600]^D$</td>
</tr>
<tr>
<td>Levy</td>
<td>$f_4(x) = \sin^2(\pi y_1) + \sum_{i=1}^{D} (y_i - 1)^2 \left[ 1 + 10 \sin^2(\pi y_{i+1}) \right] + (y_D - 1)^2$, $y_i = 1 + \frac{\sqrt{D}}{4}$</td>
<td>MN</td>
<td>$[-10, 10]^D$</td>
</tr>
<tr>
<td>Penalized1</td>
<td>$f_5(x) = \frac{1}{D} \left( 10 \sin(\pi y_1) + \sum_{i=1}^{D} (y_i - 1)^2 \left[ 1 + 10 \sin^2(\pi y_{i+1}) \right] + (y_D - 1)^2 \right) + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$, $y_i = 1 + \frac{x_{i+1}}{4}$, $u(x, a, k, m) = \begin{cases} k(x_i - a)^m; &amp; x_i &gt; 0 \ 0; &amp; -a &lt; x_i &lt; a \ k(-x_i - a)^m; &amp; x_i &lt; 0 \end{cases}$</td>
<td>MN</td>
<td>$[-50, 50]^D$</td>
</tr>
<tr>
<td>Penalized2</td>
<td>$f_6(x) = 0.1 \left{ \sin^2(3\pi x_1) + \sum_{i=1}^{D} (x_i - 1)^2 \left[ 1 + 10 \sin^2(3\pi x_i) \right] \right} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$</td>
<td>MN</td>
<td>$[-50, 50]^D$</td>
</tr>
<tr>
<td>Quartic</td>
<td>$f_7(x) = \sum_{i=1}^{D} i x_i^4$</td>
<td>US</td>
<td>$[-1.28, 1.28]^D$</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>$f_8(x) = \sum_{i=1}^{D} (x_i^2 - 10 \cos(2\pi x_i) + 10)$</td>
<td>MS</td>
<td>$[-5.12, 5.12]^D$</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>$f_9(x) = \sum_{i=1}^{D-1} \left[ 100 (x_i^2 - x_{i+1})^2 + (x_{i+1} - 1)^2 \right]$</td>
<td>MN</td>
<td>$[-50, 50]^D$</td>
</tr>
<tr>
<td>Schwefel P2.22</td>
<td>$f_{10}(x) = \sum_{i=1}^{D}</td>
<td>x_i</td>
<td>+ \prod_{i=1}^{D}</td>
</tr>
<tr>
<td>Schwefel P1.2</td>
<td>$f_{11}(x) = \sum_{i=1}^{D} \left( \sum_{j=1}^{i} x_j \right)$</td>
<td>UN</td>
<td>$[-100, 100]^D$</td>
</tr>
<tr>
<td>Schwefel P2.21</td>
<td>$f_{12}(x) = \max {</td>
<td>x_i</td>
<td>, 1 \leq i \leq n}$</td>
</tr>
<tr>
<td>Sphere</td>
<td>$f_{13}(x) = \sum_{i=1}^{D} x_i^2$</td>
<td>US</td>
<td>$[-100, 100]^D$</td>
</tr>
<tr>
<td>Step</td>
<td>$f_{14}(x) = \sum_{i=1}^{D} \left(</td>
<td>x_i + .5</td>
<td>\right)^2$</td>
</tr>
</tbody>
</table>