Experiments on multi-swarm elitist version of particle swarm optimization for numerical functions

Corina Dimitriu
Faculty of Computer Science
"Alexandru Ioan Cuza" University of Iași
Iași, Romania
corina.dimitriu.01@gmail.com

Abstract—**This paper investigates the influence on behaviour and results of applying a combination between two particle swarm optimization directions, the multi-swarm variant and the rather elite oriented one, while additionally employing particles' classification, Gaussian and Cauchy position sampling, adaptive group size and ring-shaped neighborhood topology, over a popular benchmark of mathematical functions: Rastrigin's, Michalewicz's, Griewangk's and Rosenbrock's. Since previous experiments already proved that PSO is prone to suffer from the curse of dimensionality, partitioning the learning process into co-evolutive information distribution within dynamic subpopulations, followed by guiding knowledge in a divide-and-conquer manner, adds social character forming strategy to the base stochastic nature of the algorithm. The results are therefore evidence of a balanced collective effort between cooperative group evolution and elitist leadership within the groups — which both show major contribution in defying the trap of local optima — towards crossing the barriers imposed by high dimensionality and challenging function landscape. Despite its narrow appearance and non-negligible complexity, the optimization trajectory explores the search space through numerically significant steps, in terms of quality and speed, even up to efficient and highly adaptive convergence towards the actual functions' minima.**

Index Terms—**Multi-swarm, evolution, elitist, subpopulation, optimization, particle, convergence, dimension**

I. Introduction

On the premise of producing high quality results in fairly time sensitive optimization scenarios, particle swarm optimization technique particularly mantains an honorable position due to its diverse variants, which are each meant to enhance the reliability and robustness of the standard base version of PSO, in diversely targeted cooperative and competitive terms. It is quite notorious that the base PSO algorithm performs poorly in complex or particular shape dependent search spaces, therefore failing to provide desirable results for high-dimensional optimization problems (reffering 10-dimensional or 30-dimensional scenarios inclusively), with possible challenging search patterns to be adopted for convergence, which makes a wide range of alternative versions fit into five different categories: adaptive versions, hybrid versions, topology-enhanced versions, multi-swarm versions and fast converging versions [1, 2].

Previous work has shown the effects of implementing and finetuning instances from each category of PSO versions and eventually mixing strategies which respond to similar stimuli in the optimization context (i.e., evolve related environments), such as random grouping and dynamically changing group size — which both respond to cooperation needs — but more valuable results were acquired when controlling behaviour learning tasks through apparently unrelated optimization keys [3, 4]. However, little research has been done on meta learning approaches for the optimal combination of previous individually developed strategies. To eloquently illustrate the improvement on the swarm's social behaviour through cooperation between strategies, X. Li and X. Yao came with the CCPSO2 algorithm [3, 4]. This is where a "cooperation for cooperation" optimization system comes into place for co-evolving subpopulations using the interactions within a diverse (but high aiming) combination of intelligence centers located among each subpopulation and exploited by each strategy in its characteristic manner.

The experiments described in this paper aim to identify a well performing, as general as possible, ensemble of strategies and, going deeper, present the dependencies between the converging and qualititative efficiency of this ensemble and the parameter choices for each component aside as well as for the whole model. In terms of numerical optimization, both cooperation and competition encourage adaptive, diversified, on target fast learning [5]. Consequently, for the purpose of building a cooperative but elitist algorithm, the following sections are built upon three topical questions:

- Can we simultaneously attain quality solutions and fast convergence?
- Given the stochastic nature of the algorithm, what is the proper amount of randomness to adopt for the position update procedure and how the current population state can relate to it?
- What is the lifecycle of the particles' individual and collective memory?

The paper logic is structured as follows. Section II details the construction of the proposed particle swarm optimization algorithm, with great emphasis on treating the background and contribution of each structure element individually. Section III covers the experiments performed with respect to the five test functions — Rastrigin's, Michalewicz's, Griewangk's and Rosenbrock's — the experiments being preceeded by empirical determination of an optimal model parameterization, with high co-evolution coefficient and reasonable degree of universality, for the employed algorithm structure in Section II. Section IV provides the experimental setup with the corresponding results and interpretation hypotheses based on analysis and generalization. Additionally, Section V presents a suite of theoretical and experimental comparisons between PSO and GA implementations. Finally, Section VI presents the conclusions which derive from the full track for (parameterized) model completion being followed in the previous sections; further directions are also briefly highlighted.

II. Particle Swarm Optimization

Particle Swarm Optimization relies on a swarm intelligence social paradigm defined by Kennedy and Eberhart in 1995, which assumes both individual and collective intelligence for the evolved population [2]. That being said, the individuals make use of simple mechanics to strike a balance between exploiting own knowledge and trusting collectively shared information, therefore adjusting their positions and velocities so as to move toward desirable final solutions.

A. Operating principles

The base version of PSO implements an easy to follow position update equation, equipped with knowledge on productive directions and randomness alike, making sure both exploration and exploitation implicitly intervene in the equation [4]:

$$\begin{aligned} v_i(t+1) = w \cdot v_i(t) + c_1 \cdot rand() \cdot (pbest_i(t) - x_i(t)) \\ + c_2 \cdot rand() \cdot (gbest(t) - x_i(t)) \\ x_i(t+1) = x_i(t) + v_i(t+1) \end{aligned} \quad (1)$$

where v_i stands for the velocity of the particle, x_i stands for its position, w is the inertia weight which saliently controls the balance bewteen exploration and exploitation and $c1$ and $c2$ denote the cognition and social constants respectively, which determine how much influence the personal best so far (i.e. $pbest$) and the global best (i.e. $gbest$) will each posses over the moving direction to be decided [1]. Last but not least, t stands for the current iteration. Sometimes the global best is replaced by $lbest$, which denotes the best position found among a certain neighborhood, or $sbest$, which qualifies the best position among a particles' subpopulation (not necesarilly comprising dimensions of neighboring individuals). It is important to notice that the population, as it is referred in PSO algorithms, terms a conglomerate of particles, one for each dimension to optimize for each individual.

In [6], van den Bergh and Engelbrecht built the foundation of a cooperative PSO version (CPSO), which was further transformed by Li and Yao into a cooperative coevolutionary (CCPSO) algorithm and, later on, an enhanced multi-strategy cooperative coevolutionary version (CCPSO2) [3, 4]. The main idea behind cooperative coevolution (CC) is to split the total number of dimensions into K dimension groups and assess each individual for fitting each group. Consequently, subindividuals are defined as the individuals' projections over the target dimensions' set of the group. Moreover, the fitness assessment implicitly handles both the integration coefficient of the subindividual within its group and within the whole swarm, since it is directly combined with best members coming from other subpopulations. As

$$dim\_size = s \cdot k \quad (2)$$

where dim_size is the number of dimensions for the search space, s is the group's size and k is the number of groups, each group may be seen as a bucket attempting to give a chance to each individual's projection over its dimensions' set, among which $\hat{y}$ is the most adapted to the fitness landscape (Fig. 2).

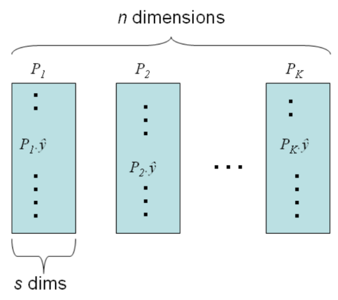

Fig. 1. Dividing the population into collaborating swarms [3, 4].

B. Proposed version of PSO

The version of PSO explored until completion in this section finds its strength in a combination between random grouping, which falls into the category of cooperative multi-swarm strategies, adaptive group size, which is specific to adaptive PSO strategies, *lbest* ring topology in the local neighborhood definition scheme and Gaussian and Cauchy-based sampling around the personal and neighborhood best respectively, which are both inspired by topology-enhanced PSO versions, and elitist particles' classification accompanied by targeted elitism, which characterize the elitist behaviour usually proposed in fast converging algorithm variants (see Section I).

As both realibility and performance are concerned, the proposed algorithm is highly adaptive to large search spaces, since it decomposes the population and then recomposes most well fitted particle groups, but it is also robust to narrow variations in the fitness landscape, due to its dynamic neighborhood structure and elite oriented flow. Nevertheless, the employed grouping strategy has its roots in diferential evolution strategies and attempts to separately optimize the interactions between dependent variables, so that they can then benefit from an elitistly structured optimization component, whose close relationship with the division mechanism enables the whole process's working flow to be summarized in a pretty easy to follow pseudocode [Algorithm 1].

1) Random grouping: Apart from the group forming procedure, which results in k groups comprising s dimensions each and covers all the dimensional complexity of the search space, it was proven an additional strategy which randomly and frequently changes the groups' composition might also improve the quality of solutions targeted by the algorithm [3]. The issue this mechanism adresses is the inherent possibility for

Algorithm 1: The pseudocode of the proposed PSO algorithm. $P_j.x_i$ stands for the current position of the ith population subindividual as projected over the jth swarm, while $P_j.y_i$ denotes the best memorised position of the same subindividual and $P_j.\hat{y}'$ represents the swarm's last iteration's exclusive best position. Additionally, $P_j.\hat{y}$ stands for the global best among all iterations and $best(j,z)$ replaces the jth group in the swarm's so far optimum positions: ($P_1.\hat{y}'$, $P_2.\hat{y}'$,..., $P_{j-1}.\hat{y}'$, z, $P_{j+1}.\hat{y}'$,..., $P_k.\hat{y}'$). Complete explanation of the detailed significance and notations for each line is provided within sections II.B.1, II.B.2, II.B.3, II.B.4, II.B.5, II.B.6.

```
Initialize swarms to fit the n dimensions ($n = k \cdot s$);
repeat
    if $iteration > 1$ and $no\_fitness\_improved(iteration - 1)$ then
        $s \leftarrow random(\{group\_sizes\_set\})$;
    end
    $permutate(swarms)$;
    for swarm in $[1...K]$ do
        for particle in $[1...s]$ do
            if $fitness(best(j, P_j.x_i)) < fitness(best(j, P_j.y_i))$ then
                $P_j.y_i \leftarrow P_j.x_i$;
            end
            if $fitness(best(j, P_j.y_i)) < fitness(best(j, P_j.\hat{y}'))$ then
                $P_j.\hat{y}' \leftarrow P_j.y_i$;
            end
            if $fitness(best(j, P_j.\hat{y}')) < fitness(best(j, P_j.\hat{y}))$ then
                $P_j.\hat{y} \leftarrow P_j.\hat{y}'$;
            end
        end
        Update positions through elitist Cauchy/Gaussian-based sampling (5);
    end
until convergence or maximum number of iterations;
```

the highly dependent variables to be placed in different groups at the initialization stage and hence encounter difficulties in updating their position accordingly for nonseparable scenarios. Therefore, at each iteration, each group's components are uniformly sampled until meeting the same treshold, s, for the number of members; this translates into forming a new permutation of the dimensions' indexes and then separating it into k components. It can be statistically demonstrated that, as the number of performed shuffles increases, the probability of placing two interacting variables and co-evolving them at least once into the same component becomes higher; for only one independent iteration, the probability starts at 0.1, but if considering n groupings the probability can be computed after the following formula:

$$\begin{aligned} P(x \geq 1) &= p(1) + p(2) + p(3) + ... + p(n) \\ &= 1 - p(0) \\ &= 1 - \binom{n}{k} \cdot (0.1)^0 \cdot (1 - 0.1)^n \end{aligned} \tag{3}$$

where x denotes the number of positive conclusions from observing two interacting variables being located within the same component, keeping in mind that a binomial distribution is being followed. Therefore, after no more than 50 iterations the probability reaches 0.9948 for any arbitrarily selected two variables within the search space, this result by itself supporting the implementation of random grouping within the proposed PSO solution.

2) Adaptive group size: Since the group size has a salient influence over the results, just like the population size, a constant choice for the s value might start losing its solution improvement potential at some point during the algorithm's execution [4]. Taking advantage of the fact that a sufficient number of coevolutionary cycles are performed, a set of empirically determined high impact group sizes may be proposed and a new group size may be chosen uniformly at random from that list after certain execution periods. Varying the group size forces the exploration process to traverse a wider range of subpopulation co-evolutive behaviours and the exploitation component to admit a higher probability of including dependent variables within the same groups. In the context of this paper, the changing criterion followed the example given by Li and Yao in their description of the CCPSO2 algorithm, which imposes the adaptation to happen after those iterations which do not modify the fitness value [4]. In other words, the group size remains unchanged for whatever long it still helps in producing better fitness values. However, the supplied group sizes set is of great importance in tuning that exploratory behaviour which aims at capturing the proper amount of interactions between variables and has been experimentally demonstrated to influence the algorithm's way of reacting to local optima (see Section III.B). Popular choices for the group sizes within this paper were proper and improper divisors of the dimensional complexity number which are greater than 5.

*3) **lbest** ring topology:* The proposed algorithm implements an additional exploitation layer through the *lbest* ring topology, which replaces the traditional role of *gbest* in the canonical PSO version with local neighborhood based update rules. Despite aiming intentionally for slower convergence and therefore passing through more possible local optima, local neighborhood strategies also facilitate methodically escaping most of the local optima. It is the slower convergence which enables the quality performance of the algorithm to tackle reliability issues in multimodal functions. In particular, the *lbest* ring topology takes a static ordering of the population and considers the implicit individuals located on the left and right side respectively as neighbors, while additionally enforcing the ordering with a rule that declares the left neighbor of the first individual as the last one and the right neighbor of the last individual as the first individual. As the description of targeted elitism will clarify in the following, the use of *gbest* is not completely removed from the algorithm's structure, but it is scaled to the last iteration's exclusive optimum.

4) Gaussian and Cauchy-based sampling: It was decided for this algorithm version to not make use of the classical velocity vector, but rather elitistly sample around the three categories of so far discovered optima. The base version of Gaussian/Cauchy sampling, proposed by tpology-enhanced strategies in their mostly general form and particularly developed within the CCPSO2 algorithm, is firstly presented, so as to introduce the working principle behind, following that the complete update schema will be revealed within the targeted elitism description as a mathematical equation which formally represents the main contribution of this paper.

$$x_{i,d}(t+1) = \begin{cases} y_{i,d}(t) + \mathcal{C}(1) \cdot |y_{i,d}(t) - \hat{y}'_{i,d}(t)| & p \leq 0.5 \\ \hat{y}'_{i,d}(t) + \mathcal{N}(0,1) \cdot |y_{i,d}(t) - \hat{y}'_{i,d}(t)| & p > 0.5 \end{cases} \tag{4}$$

Equation (4) denotes the particle's personal best position through $y_{i,d}$ and the particle's current position through $x_{i,d}$, where the (i,d) tuple specifies the individual and dimension which uniquely identify the particle. Furthermore, $\hat{y}'_{i,d}$ denotes the *lbest* value for the particle considered, meaning the best neighbor (the particle itself included) following the ring topology, while $\mathcal{C}(1)$ stands for a number generated according to a Cauchy distribution with "effective standard deviation" 1 and $\mathcal{N}(0,1)$ stands for a number generated according to a standard Normal distribution (it is instructive to notice that the standard deviation for the Gaussian distribution and the "effective standard deviation" for the Cauchy distribution are alike).[1] The equation's overall effect targets the dissolution of premature convergence through equally combining (p denotes the probability to apply Cauchy sampling) the Gaussian and Cauchy sampling methods. It is widely accepted that the Cauchy and Gaussian distribution used together prove stronger exploration capabilities than the Gaussian distribution alone, since the use of the Gaussian distribution alone tends to gradually diminish its dispersion capacity. In fact, this is one reason why the Cauchy distribution was considered well suited for exploring around the personal best and the Gaussian distribution's properties are exploited to sample around the neighborhood best.

5) Elitist particles' classification: Until now the proposed algorithm entirely pursues the coevolutionary cooperative paradigm imposed by the CCPSO2 version. The combination of multi-swarm and elitist behaviour has not intensively been studied so far, especially not with the purpose to improve the multi-swarm strategy's performance. Chen introduced a fitness-based particles classification system where each particle may fall within one of the following three categories: *fair* particle, if its fitness is located around the average fitness (between two established margins, $aver1$ and $aver2$), *bad* particle, if its fitness is located below an inferior margin, $aver1$, *good* particle, if its fitness is located above a superior margin, $aver2$ [16]. Employing particles' classification boosts the algorithm's stability and helps deciding individual exploration/exploitation ratios for each particle aside (see Section II.B.6), the intuition on the way this method treats premature convergence being that all particles are attempted to be gathered into the first category, but the fitness average and the two $aver1$ and $aver2$ boundaries are constantly changing at the same time depending on the particles' movement.

[1]The Cauchy distribution itself does not originally have mean neither variance [7]. Here the standard form for the Cauchy is used.

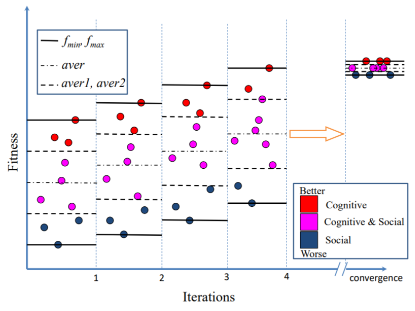

Fig. 2. Particles' classification schema [5].

6) Targeted elitism: Targeted elitism strongly relies on the particles' classification system integrated by the algorithm and it refers to adapting the update equation for each particle aside depending on the way it is classified according to its fitness value [5]. Since the use of any of the three so far discovered best solutions alone within the update equation, without any reasonable conditioning enabled, might be easily trapped by local minima, the Gaussian/Cauchy sampling system is changed as follows:

$$x_{i,d}(t+1) = \begin{cases} y_{i,d}(t) + \mathcal{C}(1) \cdot |y_{i,d}(t) - \hat{y}'_{i,d}(t)| & p \leq 0.5 \\ y_{i,d}(t) + \mathcal{N}(0,1) \cdot |y_{i,d}(t) - \hat{y}'_{i,d}(t)| & p > 0.5 \end{cases}$$
$$avg + aver2 \cdot avg \leq fitness \tag{5a}$$

$$x_{i,d}(t+1) = \begin{cases} \hat{y}'_{i,d}(t) + \mathcal{C}(1) \cdot |y_{i,d}(t) - \hat{y}'_{d}(t)| & p \leq 0.5 \\ \hat{y}'_{i,d}(t) + \mathcal{N}(0,1) \cdot |y_{i,d}(t) - \hat{y}'_{d}(t)| & p > 0.5 \end{cases}$$
$$avg - aver1 \cdot avg \geq fitness \tag{5b}$$

$$x_{i,d}(t+1) = \begin{cases} y_{i,d}(t) + \mathcal{C}(1) \cdot |y_{i,d}(t) - \hat{y}'_{i,d}(t)| & \\ \quad +\mathcal{C}(1) \cdot |y_{i,d}(t) - \hat{y}'_{d}(t)| & p \leq 0.5 \\ \hat{y}'_{i,d}(t) + \mathcal{N}(0,1) \cdot |y_{i,d}(t) - \hat{y}'_{d}(t)| & \\ \quad +\mathcal{N}(0,1) \cdot |y_{i,d}(t) - \hat{y}'_{i,d}(t)| & p > 0.5 \end{cases}$$
$$avg - aver1 \cdot avg \leq fitness \leq avg + aver2 \cdot avg \tag{5c}$$

where the newly added notation, $\hat{y}'_{d}(t)$, represents the swarm's last iteration exclusive optimum for the dth dimension and avg obviously denotes the average fitness of the whole swarm.

Targeted elitism boosts convergence, which is what makes this strategy fall into the fast converging PSO versions category. The idea which inspired this method's development and refinement proposes more salient and adaptive guiding of the particles, determining the *good* particles to speed down and perform exploitation around their personal best position, the *bad* particles to speed up and explore quicker to catch up with the swarm's average, and the *fair* particles to keep performing both exploration and exploitation, but in a slightly different manner than the one employed by the classical Cauchy/Gaussian sampling.

This elitist sampling system represents the most important contribution that this paper has generated and has experimentally proved, as it the most eloquent way of illustrating the quality improvement of the two elitist components (particles' classification and targeted elitism) being added over the multi-swarm framework. Needless to say, the effect of employing targeted elitism was crucial for optimizing the Rosenbrock's function (Fig. 3). Moreover, an additional classification layer for so-called "hopeless particles" was implemented with the purpose to permanently avoid premature convergence for the 30-dimensional subset of the benchmark functions. This layer enables particles which have not escaped the *bad* particles category for a big number of iterations to benefit from a "restart" and be placed in the swarm's last iteration exclusive optimum. The number of iterations until "restart" is adaptively computed depending on the number of executed iterations, which also provides the final version of the algorithm with a time-varying coefficient:

$$\theta = \frac{no\_iterations - run}{no\_iterations} \cdot hc \tag{6}$$

where $theta$ denotes the treshold for executing the "restart" operation and hc represents the "hopeless" coefficient, while $no_iterations$ denotes the maximum number of iterations to be executed until hoped convergence and run indexes the current iteration. Experiments proved an optimal value for hc is 50.

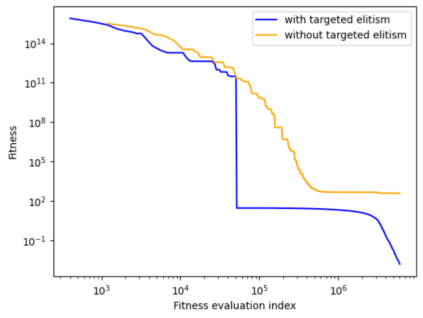

Fig. 3. Effects of adding elitism on Rosenbrock's function.

7) Fitness: In the context of numerical optimization provided in this paper, the fitness translates as the function to be minimized by the value in the global best position's coordinates. It is straightforward this function actually defines the intrinsic target of the search mechanism described in this section and attributes its efficiency to the particles' cooperative adaptation to the function's landscape. Section III exposes all the equations of the fitness functions under the label of test functions.

III. Experiments

A. Model parameterization

Apart from a strong learning ensemble, the model's optimization capability resides in the values and dynamics chosen for its parameters. Therefore model parameterization is defined as the process of filling the hyperparameter choices in the ensemble's learning skeleton in such a way that precise or nearly precise optimal behaviour is achieved.[2] In the context of the PSO version proposed within this paper, the model parameterization process is firstly segregated into the optimization of values and dynamycs for five individual components: population size (pop_size), adaptive group size (s), Cauchy/Gaussian-based sampling probability (p), average fitness percentage boundaries ($aver1$, $aver2$) and targeted elitism "hopeless" coefficient (hc), each component being optimized in an isolated framework within Sections III.B, III.C, III.D, III.E and III.F. The isolated framework is defined with respect to the parameter to be optimized and assumes its variation while keeping the other hyperparameters at fixed behaviour (optimal or nonoptimal). The final interactive framework, in which all the isolated hyperparameters' optimization results are let to vary together, create dependencies and be adjusted accordingly, is provided within Section III.G.

B. Population size

The population size, denoted by s, is likely to demonstrate similar performance when being set to lower or higher values alike, in terms of the necessary amount of time supplied for the algorithm to eventually converge. As the experiments depict, this parameter does not cause major performance leaks or enhancements when small-scale modifications are applied to it [Fig. 4]. A neutral final value of 100 for this parameter was decided according to the performed experiments.

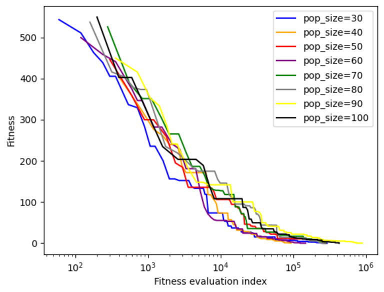

Fig. 4. Experiments on varying the population size (Rastrigin's).

C. Adaptive group size

As far as discovering and exploiting variable interactions is concerned, implementing group size dynamycs definitely revolutionizes the algorithm's performance in both quantitative and qualitative terms. In other words, not only adaptive

[2]The significance of optimal and nearly optimal behaviour varies depending on the problem's conditioning environment and the function to optimize.

group size explodes convergence speed compared to a static group size, but it also sensitive to premature convergence, both positively and negatively [Fig. 7]. That being said, the most suitable choice for this parameter (in relation with the targeted elitism) with respect to the 30-dimensional case proved to be $\{5, 6, 10, 15, 30\}$, as determined with most significant contribution from the Rosenbrock's function in particular. The 100-dimensional case was generally solved by the $\{5, 10, 20, 25, 50, 100\}$ set, with the additional specification that Michalewicz's function was slightly better handled by the $\{5, 10, 20, 25, 50\}$ set. For the 2-dimensional case only, a constant size of 2 produced better results than its dynamically changing $\{1, 2\}$ alternative, the latter even exposing the algorithm at great risk of missing the global optimum. Too many or too little possibilities for the group size with respect to the number of dimensions may cause instability, which is why an amount of 5 possibilities was considered appropriate for 30 dimensions.

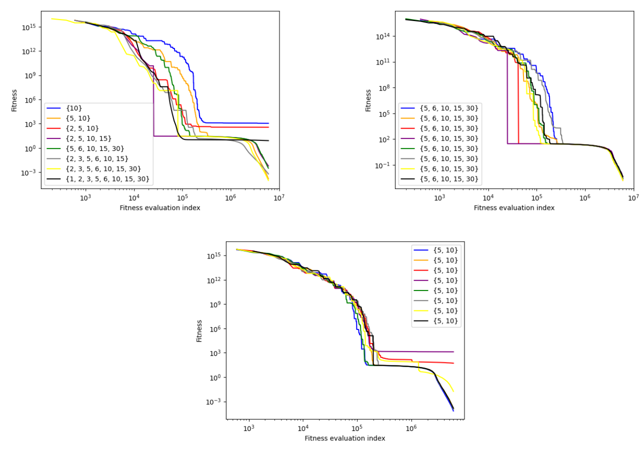

Fig. 5. Experiments on varying the group sizes set (Rosenbrock's).

D. Cauchy/Gaussian-based sampling probability

The Cauchy/Gaussian-based sampling probability, denoted by p, has to do with both exploration-exploitation and *lbest-pbest* influential proportions and can easily generate even weaker convergence if not providing enough exploration or dramatically slow down the search if not encouraging enough exploitation. Naturally following, the 0.5 candidate proves itself suitable for this parameter according to the performed experiments.

E. Average fitness percentage boundaries

In order for the average fitness percentage boundaries ($aver1$ and $aver2$) to properly exercise their influence, not too low inferior boundary has to be provided so that quite well fitted subindividuals won't pass as less adapted and therefore miss their right to wider exploration, but not too high superior boundary has to be applied at the same time so that the algorithm won't forcefully impose exploitation at the expense of unnecessarily slowing down exploration. A final value of 0.1 for both parameters was decided according to the performed experiments [Fig. 6].

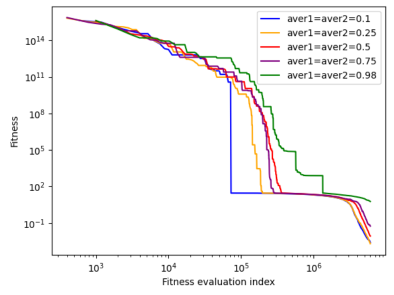

Fig. 6. Experiments on varying the average fitness percentage boundaries (Rosenbrock's).

F. Targeted elitism's "hopeless" coefficient

Since this parameter was introduced while studying convergence for the Rosenbrock's function, its dynamics is directed towards improving performance on this function's particular kind of narrow fitness landscape. However, the "hopeless" coefficient, denoted by hc, is likely to adaptively generalize its base usage to the other test functions (Fig. **??**) and speed up local optima escaping if employed. A highly optimal value for this parameter (highly dependent on the group sizes set) would be 10 decided according to the performed experiments.

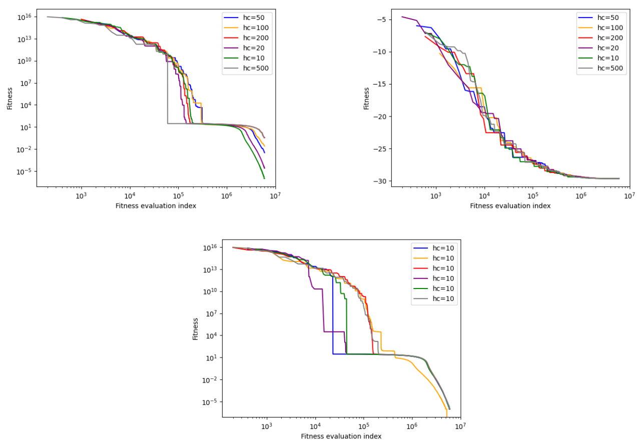

Fig. 7. Experiments on varying the group sizes set (Rosenbrock's (left and center), Michalewicz's (right)).

G. Final framework

As there is no free lunch, each experiment is repeated 30 times and statistical metrics are therefore inferred. An experiment is defined through the test function, the number of dimensions for the search space, the population size, the group sizes set (which serves for listing the possibilities of uniformly choosing the group size) and the number of successive iterations (fitness evaluations).

A maximum number of fitness evaluations was used as a more viable stopping criterion than the number of iterations, since the number of groups within each iteration varies depending on the selected group size. Consequently, the number of iterations is only an approximation of the canonical number of iterations (which would be 1000 assuming one iteration comprises $number\_of\_dimensions \cdot pop\_size$ personal best and position updates), named *standard number of iterations* in what follows, which is equivalent to itself multiplied by $number\_of\_dimensions \cdot pop\_size \cdot fe$, where fe denotes the number of fitness evaluations per non-standard iteration ($30 \cdot 100 \cdot 2 \cdot 1000 = 6000000$ fitness evaluations for the 30-dimension scenario with 100 as the population size, providing 2 fitness evaluations happen for each group–individual pair — one for replacing the group's positions with the updated individual's positions and one for replacing them with the individual's known personal best positions so far).

The experiments were run for 1000 standard iterations (with the variant of approximately 1700 for the Rastrigin's function test case) but the majority of test scenarios reached perfect convergence earlier for each of the runs, therefore the average number of fitness evaluations and the average number of necessary standard iterations were also recorded. The population size was set at 100 candidates, no matter the test function or the number of dimensions, even though a population of 30 individuals demonstrates comparable performance (see Section III.A). The proposed group sizes set was decided at $\{5, 6, 10, 15, 30\}$ for best results within the 30-dimension experiments and $\{5, 10, 20, 25, 50, 100\}$ within the 100-dimension ones (except for Michaleiwcz's which eliminates the last element). For the 2-dimension experiments the group sizes set changes its lower bound to 2 and becomes $\{2\}$, which means for this test scenario a constant group size demonstrates similar performance.

To summarize, each test function results, for each number of dimensions aside, (the possible values for the number of dimensions being $\{2, 30, 100\}$), benefited from 30 runs, comprising up to 1000 standard iterations (or, in other words, 6000000 fitness function evaluations) each (the highest number of necessary iterations being observed within the Rosenbrock's function case, with potential to still decrease the optimum found after the last iterations, which led to increasing the maximum number of fitness evaluations to 9000000 for his test case only), executed on a Lenovo Legion 5 Pro 16ARX8 laptop, equipped with AMD Ryzen 9 7945HX processor and 32GB of RAM memory. The average running time was also recorded for all the experiments. A maximum number of iterations which approximates the maximum number of fitness iterations is also set with the purpose of updating the targeted elitism's "restart" equation (see Section II.B.6).

H. Test functions

There are four functions subject to minimization within the experiments, namely Rastrigin's function, Michalewicz's function, Griewangk's function and Rosenbrock's function, whose descriptive equations and 3-dimensional plots are provided below.

That being said, the equation of the Rastrigin's function is depicted as follows [8, Fig. 8]:

$$f_6(x) = 10 \cdot n + \sum_{i=1}^{n} (x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i)) \qquad -5.12 \le x_i \le 5.12 \tag{7}$$

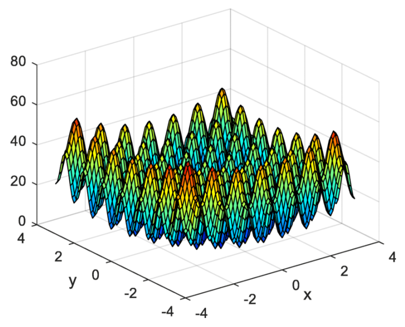

Fig. 8. Rastrigin's function plot [12].

The equation of the Michalewicz's function [9, Fig. 9]:

$$f_{12}(x) = -\sum_{i=1}^{n} \sin(x_i) \cdot \left(\sin\left(\frac{i \cdot x_i^2}{\pi}\right)\right)^{2 \cdot m} \qquad 0 \le x_i \le \pi, \quad m = 10 \tag{8}$$

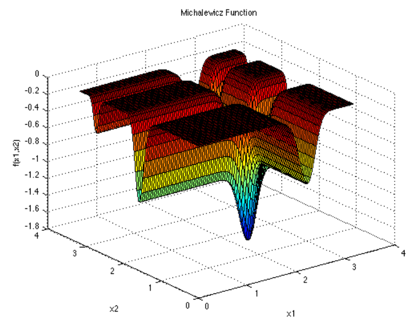

Fig. 9. Michalewicz's function plot [13].

The equation of the Griewangk's function [10, Fig. 10]:

$$f_8(x) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \qquad -600 \le x_i \le 600 \tag{9}$$

The equation of the Rosenbrock's function [11, Fig. 11]:

$$f_2(x) = \sum_{i=1}^{n-1} 100 \cdot \left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2 \qquad -2048 \le x_i \le 2048 \tag{10}$$

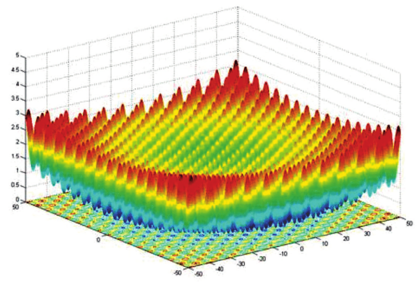

Fig. 10. Griewangk's function plot [14].

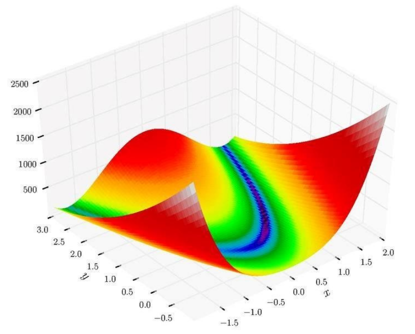

Fig. 11. Rosenbrock's function plot [15].

IV. Results

The results obtained within the previously described framework are summarized in tables recording the minimum, maximum, mean and standard deviation amongst the found minima. Additionally, average running time and necessary iterations statistics were computed. Needless to say, each run resulted in one minimum point, which is the best ever generated candidate across all generations. In addition, statistics regarding each configuration on 30 dimensions were included through convergency plots, picturing the dependency between the reached generation and the optimum trajectory, provided with explanations capturing the exploration-exploitation dichotomy in relation with the fitness properties in one sample run.

For Rosenbrock's function, the maximum number of standard iterations is firstly set to 1000 and statistics are computed, but since it was observed there is still better quality convergence potential, the maximum number of fitness evaluations is increased to 9000000 within a second round of experiments provided for this function only (Fig. IX).

TABLE I
Numerical quality performance assessment of proposed PSO version on Rastrigin's function

Dimensions	Statistical results			
	Min	*Mean*	*StDev*	*Max*
2	0.00000	0.00000	0.00000	0.00000
30	0.00000	0.00000	0.00000	0.00000
100	0.00000	0.00000	0.00000	0.00000

Placing the collective tendency on the cutting board first, the results determined within this final framework are able

TABLE II
Time for convergence performance assessment of proposed PSO version on Rastrigin's function

Dimensions	Statistical results	
	Avg running time (sec.)	*Avg standard iterations*
2	0.00000	32.733333
30	0.53333	56.20666
100	3.66667	60.20066

TABLE III
Numerical quality performance assessment of proposed PSO version on Michalewicz's function

Dimensions	Statistical results			
	Min	*Mean*	*StDev*	*Max*
2	-1.80130	-1.80130	0.00000	-1.80130
30	-29.63088	-29.63088	0.00000	-29.63088
100	-99.41186	-99.35173	0.00208	-99.26905

TABLE IV
Time for convergence performance assessment of proposed PSO version on Michalewicz's function

Dimensions	Statistical results	
	Avg running time (sec.)	*Avg standard iterations*
2	0.00000	9.95000
30	17.13333	1000.00000
100	152.66667	1000.00000

TABLE V
Numerical quality performance assessment of proposed PSO version on Griewangk's function

Dimensions	Statistical results			
	Min	*Mean*	*StDev*	*Max*
2	0.00000	0.00000	0.00000	0.00000
30	0.00000	0.00000	0.00000	0.00000
100	0.00000	0.00000	0.00000	0.00000

TABLE VI
Time for convergence performance assessment of proposed PSO version on Griewangk's function

Dimensions	Statistical results	
	Avg running time (sec.)	*Avg standard iterations*
2	0.03333	103.71667
30	0.40000	38.53888
100	2.96666	35.48800

TABLE VII
Numerical quality performance assessment of proposed PSO version on Rosenbrock's function with stopping criterion set to 1000 standard iterations

Dimensions	Statistical results			
	Min	*Mean*	*StDev*	*Max*
2	0.00000	0.00000	0.00000	0.00000
30	0.00096	0.00358	0.00002	0.02741
100	0.00000	4.68431	69.89910	34.37446

TABLE VIII

TIME FOR CONVERGENCE PERFORMANCE ASSESSMENT OF PROPOSED PSO VERSION ON ROSENBROCK'S FUNCTION WITH STOPPING CRITERION SET TO 1000 STANDARD ITERATIONS

Dimensions	**Statistical results**	
	Avg running time (sec.)	***Avg standard iterations***
2	0.06666	402.90000
30	15.36666	1000.00000
100	130.96666	1000.00000

TABLE IX

NUMERICAL QUALITY PERFORMANCE ASSESSMENT OF PROPOSED PSO VERSION ON ROSENBROCK'S FUNCTION WITHOUT STOPPING CRITERION

Dimensions	**Statistical results**			
	Min	***Mean***	***StDev***	***Max***
30	0.00000	0.00000	0.00000	0.00000

TABLE X

TIME FOR CONVERGENCE PERFORMANCE ASSESSMENT OF PROPOSED PSO VERSION ON ROSENBROCK'S FUNCTION WITHOUT STOPPING CRITERION

Dimensions	**Statistical results**	
	Avg running time (sec.)	***Avg standard iterations***
30	26.29999	1726.32150

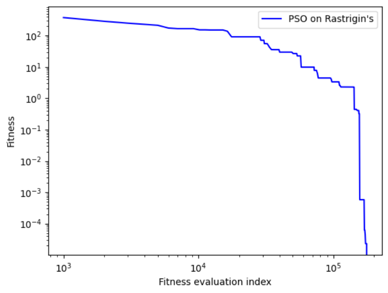

Fig. 12. 30-dimension Rastrigin's PSO fitness function plot [Fig. I].

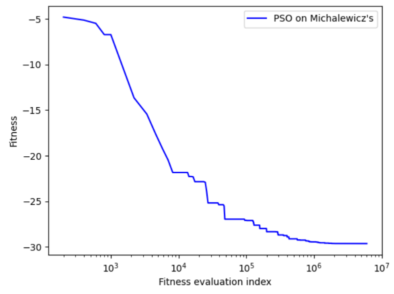

Fig. 13. 30-dimension Michalewicz's PSO fitness function plot [Fig. III].

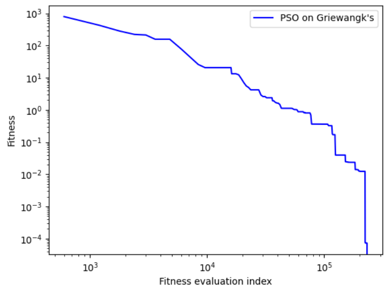

Fig. 14. 30-dimension Griewangk's PSO fitness function plot [Fig. V].

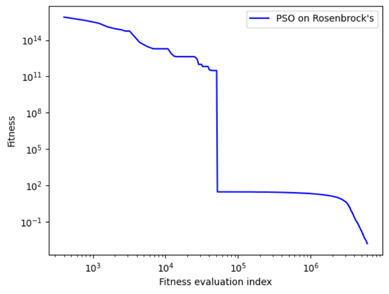

Fig. 15. 30-dimension Rosenbrock's PSO with max_fit_eval stopping criterion fitness function plot [Fig. VII].

to explain the model parameterization from a wider experimental and statistical point of view than the previous isolating framework employed for applying individual parameter changes, as the compliance with the parameter interactions is aditionally assessed. Furthermore, increasing the problem's dimensionality challenges the general framework's stability and introduces premature convergence issues in some of the 100-dimensional test cases, which makes the particular, function dependent choices intervene for later on refining, as it is the Michalewicz's case for the group size set. Moreover, the experiments on the Rosenbrock's test function determined the employment of "hopeless particles" detection, which exercised benefits on other test cases as well, in terms of quicker and firmer local optima escaping.

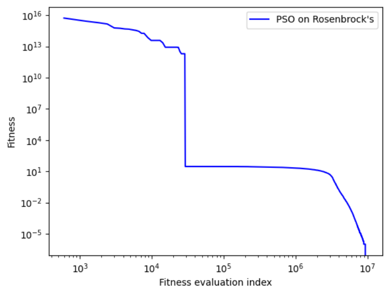

Fig. 16. 30-dimension Rosenbrock's PSO without max_fit_eval stopping criterion fitness function plot [Fig. IX].

On the individual analysis of the fitness function's behaviour, perhaps the most stringent observation regarding the used number of iterations is that while the general tendency assigns less iterations to lower dimensionalities and more iterations to higher dimensional search spaces, this rule is being defied by the Griewankg's test case, which needs more iterations for lower dimensional spaces and therefore gives an insight into the group sizes set's large scale usability; from all parameters to be set, it is the hardest to achieve a generally applicable group sizes set, with the right number of choices and the right values for those choices, since capturing the dependencies between variables is highly related to the test function and becomes more tangled as the problem's dimensionality increases. Rastrigin's function proved itself to be the most obedient function, since the vast majority of the parameter choices meeting reasonable performance requirements for the other functions quickly reached actual optimum for this function in particular.

All in all, a larger benchmark of test functions could be employed in the future in order to decide upon some inference rules for parameter choices based on the function's properties. As far as the current benchmark is concerned, the parameterization process is not exactly brain surgery, since the model appears to be quite robust to the host environment. It also gets infinitesimally close to the actual optimum, at the end of each run, for three out of four test functions on the 30-dimensional search space (Michalewicz being the exception, but still approaching the actual optimum with an approximate $2 \cdot 10^{-2}$ error), which demonstrates consistent and ordered behaviour. However, as the search space's dimensionality increases, the actual parameterization strategy, focused on trial and error aggregates, would benefit from a more methodical approach at least partially relying on the function's properties.

V. Comparison with genetic algorithms

From a computational optimization perspective, GA and PSO share many common characteristics, as they both rely on heuristically evolving candidate solutions, but they also compete against each other to dominate next overstepping of the already known performance limits, in terms of both reliability and computational cost. Both strategies are of great interest for the benchmark functions this paper kept referring to, which encourages a comparison between them to be provided within the current section, from a theoretical perspective, on the natural behaviour simulating side, but also with respect to an experimental perspective, on the results interpreting side.

A. *Theoretical comparison*

The theoretical comparison between the two algorithms targets the general similarities and differences in the way they approach natural behaviour simulation with the common purpose to search for optimum within a given search space. Starting from the causality factors which triggered their continuous improvement over years, both algorithms have to deal with the same main issues: local optima, premature convergence and high dimensionality related weak convergence.

Both strategies start with a base population (uniformly sampled for the majority of times) and evolve it towards looking for optimal solution(s) while also implying a certain amount of randomness [1, 17]. However, PSO is a social behaviour oriented strategy, whereas genetic algorithms are more interested in the natural inheritance and strive for survival, which makes the two algorithms tackling the concept of evolution from two distinct points of view. PSO does not define genetic operators, such as mutation or crossover, for biasing the information sharing mechanism, but the particles internally update their velocity towards various patches of the search space, depending on the information shared by their best neighbors; at the same time, genetic algorithms do not give their best individuals such power to rule over the entire population's update, but rather spreads the power so that the population remains cohesive and moves further "as a group". That being said, the strengths of particle swarm optimization reside in one-way information sharing and implicit cooperation towards far to near information distribution, which both fit for time and convergence efficiency, if paying attention to local minima, while the key capabilities of genetic algorithms are more likely to be mutual information sharing and parallel evolution, which strive for patiently exploring the search space in more detail and, therefore, outputing higher solution quality. This is why, in a wide variety of test cases (although there are plenty of exceptions as well), genetic algorithms might be considered more suitable for high precision, narrow and frequently twisted function landscapes, whereas PSO is more likely to be adopted under time pressure and high dimensionality conditioning, due to its favorable convergence speed.

As the structure employed by the two methods diverges ever since establishing the prevailing natural behavior they opt for, metrics such as accuracy and convergence speed highly depend upon the problem's particular requirements as well. However, since there is also a common heuristic front, four 30-dimensional benchmark functions were chosen as landmark for discussing their comparative performance in numerical optimization tasks within Section V.B.

B. *Experimental comparison*

Apart from the problem requirements, the optimization applied to the algorithm itself plays a vital role in addressing issues such as local optima and, inevitable to follow, premature convergence. Here two particular versions belonging to each of the strategies were implemented and intensively tested so as to confirm, deny and make exception to the theoretical backgorund aspects provided within Section V.A.

The PSO version used within the experiments is the one proposed in Section II of this paper and the genetic algorithm is an adaptive version of the classical binary solution, with tournament selection and the following update equations for mutation (11) and crossover (12) probabilities (p_m and p_c, respectively):

$$p_m = \begin{cases} \frac{(p_{m1}-p_{m2})\cdot(f_{avg}-f)}{f_{avg}-f_{min}+\epsilon} & f \leq f_{avg} \\ p_{m1} & f > f_{avg} \end{cases} \tag{11}$$

where p_{m1} and p_{m2} take predefined values ($p_{m1} = 0.1$ and $p_{m2} = 0.001$) and establish boundaries for the mutation probability, f is the fitness value of the candidate, f_{min} is the optimum fitness value across all candidates and f_{avg} is the average fitness of the current population.

$$p_c = \begin{cases} \frac{(p_{c1}-p_{c2})\cdot(f_{avg}-f')}{f_{avg}-f_{min}+\epsilon} & f' \leq f_{avg} \\ p_{c1} & f' > f_{avg} \end{cases} \tag{12}$$

where p_{c1} and p_{c2} take predefined values ($p_{c1} = 0.9$ and $p_{c2} = 0.6$) and establish boundaries for the crossover probability, f' is the fitness value of the more adapted candidate within the pair, f_{min} is the optimum fitness value across all candidates and f_{avg} is the average fitness of the current population.

The figures capturing the experiment's results are organized as follows. The first subfigure represents the convergence plot for the PSO algorithm, the second one depicts the convergence of the genetic algorithm and the third subfigure visually discriminates between the two numerical average optima obtained by the strategies.

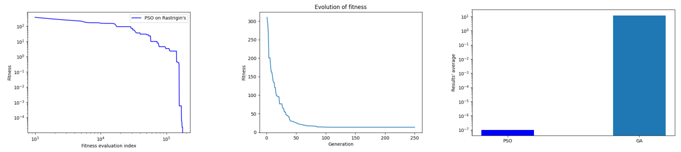

Fig. 17. Comparison on Rastrigin's function.

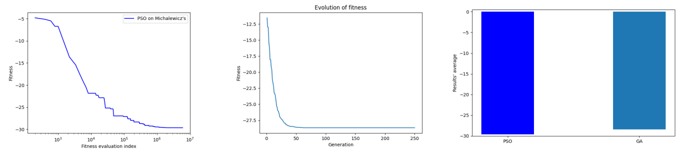

Fig. 18. Comparison on Michalewicz's function.

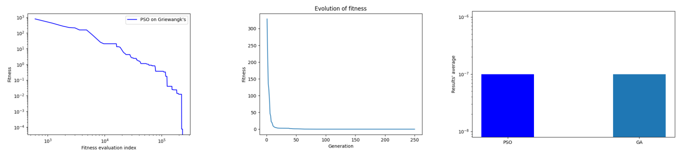

Fig. 19. Comparison on Griewangk's function.

Starting from the side by side display of each two convergence plots, one argument in favor of genetic algorithms more precisely handling the solution's quality would be their thoroughness in small step searching for the optimum, while PSO's convergence plots seem to usually imply steeper descents. However, according to its abrupt convergence within certain reduced areas from the search space, PSO demonstrates better exploitation capabilities, due to employing one-way information sharing, as stipulated. Although one would normally expect the genetic algorithm to be more stable beacuse of its similar convex curve throughout the search space — no matter the test function — the PSO algorithm obtains exact same results throughout repeated executions, whereas the standard deviation in the genetic algorithm's scenario is much more higher.

Furthermore, PSO's general quicker convergence can easily be confirmed or denied with the help of transforming the necessary number of fitness evaluations into approximate standard iterations, to which the clearly observed PSO's higher speed per iteration is added. Since the population size was set to the same value within the two strategies and one iteration technically implies applying the complete algorithm's procedure over the entire population exactly once, the PSO's amount of necessary iterations passes as dramatically smaller than the GA's executed number of iterations until convergence or comparable result in two out of three function cases. The Griewangk's function makes exception to this general behaviour and it is also fair to notice that the multi-swarm approach determines the PSO ensemble to inherently get closer to the "moving as a group" methodology — generally exploited by the GA — on pretty wide areas, so the exception happens in completely equitable conditioning. However, it appears it is in the proposed PSO version's nature to more firmly deal with the local optima and premature convergence issues, whilst the genetic algorithm remains trapped and does not reach actual minimum within the same search spaces PSO succeeds in perfectly reaching it, as it is the context of Rastrigin's function. As a consequence of its targeted elitist strategy, PSO behaves better with respect to local optima. Additional experiments showed this rule keeps being respected for more complex search spaces as well, as it is the case of Michalewicz's function on 100 dimensions. Within the context of large search spaces, the PSO's update system is reckoned to be more effective than the one belonging to the genetic algorithm, as the steep descents determined by far to near collaboration help tackling high dimensionality in a more exploitative manner. On the contrary, this exploitation-centered behaviour does not fit well with the Griewangk's function landscape, which is approached through a much more straightforward curve by the genetic algorithm.

To conclude, as long as the particular functions' set used within the experiments is concerned, PSO appears to generally traverse the search space more efficiently, with respect to all target criteria: accuracy, convergence quality, convergence speed and time performance. However, there is plenty of possibilities to improve the genetic algorithm's general performance, as it is nonetheless more prone to suffer less parameter modification if required to respond to irregular complications added over the fitness function.

VI. Conclusions

Taking into account the results of all experiments performed, there is enough emipirical evidence to conclude that model parameterization is the insufficiently praised backbone of the proposed algorithm's performance. Although there could have been no such performance without the model itself combining complex fitness landscape resistant strategies, the parameter tuning indeed has the last word in forcing more edgy time and quality performance jumps. As quality convergence is never guaranteed, let alone convergence to the actual function's minima, the model's finally decided parameters foster numerically relevant steps towards clearly improved solutions. As the numerous experiments have proved, there might be function dependent parameter choices which ultimately refine or speed up the general version of the framework, therefore it is natural for the model to question and optimize each dependency in detail, up to critically low performance differences.

Keeping the spirit of effective space exploration, future work might revolve around hybridization methodologies, which fall into the only unexplored PSO versions category within this paper. It is possible that PSO takes inspiration from genetic algorithms' working principles so as to implement mutation over elitism. In the context of the currently proposed PSO version, mutation might facilitate the comeback of the best global position within the particles' update equation, since a certain amount of diversity would be provided by mutation itself being applied to it. Moreover, parallel and distributed versions of PSO might constitute core topics in further investigation on time efficiency for even higher dimensional problems, with respect to which multi-swarm approaches have already proven their standalone efficiency.

Further directions aside, the present model builds a thoroughly optimized ensemble of some of the most competitive PSO versions, by combining their strengths in the name of optimizing reliability, robustness to local minima and convergence speed. In complete agreement with the ensemble being built upon the principle which states that "unity is strength", experiments have shown the current schema demonstrates desirable behaviour on a widely used sample of numerical functions, as implied by the conclusive benchmarks. Eventually, there is room for experiencing with methodical parameter variations and ingenious hybridizations in order to found even more effective systems suiting the properties and behaviour of the objective functions throughout the search space.

REFERENCES

[1] S.N. Sivanandam and S.N. Deepa, "Introduction to Genetic Algorithms,", Springer Berlin, Heidelberg, October 2007.

[2] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization, ", Information Sciences, January 2015.

[3] X. Li and X. Yao, "Tackling High Dimensional Nonseparable Optimization Problems By Cooperatively Coevolving Particle Swarms," 2009 IEEE Congress on Evolutionary Computation, Trondheim, Norway, May 2009.

[4] X. Li and X. Yao, "Cooperatively Coevolving Particle Swarms for Large Scale Optimization," in IEEE Transactions on Evolutionary Computation, vol. 16, no. 2, April 2012.

[5] T. Shaqarin and B. R. Noack, "A Fast Converging Particle Swarm Optimization through Targeted, Position-Mutated, Elitism (PSO-TPME)," Int J Comput Intell Syst, vol. 16, January 2023.

[6] F. van den Bergh and A. P. Engelbrecht, "A Cooperative approach to particle swarm optimization," in IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, June 2004.

[7] T. Richer and T. Blackwell, "The Levy particle swarm," in Proc. IEEE CEC, Sep. 2006.

[8] GEATbx, "Rastrigin's function 6," Available: http://www.geatbx.com/docu/fcnindex-01.html#P140_6155 [Accessed: Nov. 27, 2023].

[9] GEATbx, "Michalewicz's function 12," Available: http://www.geatbx.com/docu/fcnindex-01.html\#P204_10395 [Accessed: Nov. 27, 2023].

[10] GEATbx, "Griewangk's function 8," Available: http://www.geatbx.com/docu/fcnindex-01.html\#P160_7291 [Accessed: Nov. 27, 2023].

[11] GEATbx, "Rosenbrock's function 2," Available: http://www.geatbx.com/docu/fcnindex-01.html#P129_5426 [Accessed: Nov. 27, 2023].

[12] ResearchGate, "Plot of the Rastrigin's function for two variables," Available: https://www.researchgate.net/publication/349231806/figure/fig2/AS:990257155682305@1613107112790/Plot-of-the-Rastrigins-function-for-two-variables.png [Accessed: Nov. 27, 2023].

[13] Virtual Library of Simulation Experiments, "Michalewicz Function," Available: https://www.sfu.ca/~ssurjano/michal.png [Accessed: Nov. 27, 2023].

[14] ResearchGate, "Two variable Griewangk function," Available: https://www.researchgate.net/publication/259216728/figure/fig5/AS:454053038694406@1485266088290/Two-variable-Griewangk-function.png [Accessed: Nov. 27, 2023].

[15] ResearchGate, "Rosenbrock's function 2," Available: https://www.researchgate.net/publication/329327912/figure/fig2/AS:1078480783572992@1634141264160/Rosenbrocks-Function-in-3D.jpg [Accessed: Nov. 27, 2023].

[16] G. Chen, "Simplified particle swarm optimization algorithm based on particles classification," Sixth International Conference on Natural Computation, ICNC 2010, Yantai, August 2010.

[17] R. Cazacu,"Comparison between the Performance of GA and PSO in Structural Optimization Problems," American Journal of Engineering Research (AJER), vol. 5, 2016.

[18] E. Croitoru, Class Lecture, Topic: "Nature-inspired methods laboratory," Faculty of Computer Science Iași, Fall 2023.