Deterministic Procedures for Derivative-Free Algorithms on Multimodal Problems

March 2021

Marco Antonio Florenzano Mollinetti

Abstract

Nonlinear optimization problems that feature an uncountable amount of local optima represent a challenging class of problems where a large part of instances are so ill-conditioned that the only viable alternative is to use derivative-free algorithms. Two subfamilies of derivative-free algorithms in particular, population heuristics and direct search are among the best candidates to produce good results to those problems. Due to the impracticality of obtaining information from the curvature of the functions by means of calculation of gradients, these methods rely on randomness to guide their solutions. In light of this, this work presents several deterministic procedures to two derivative-free algorithms, the Artificial Bee Colony and the Nelder-Mead direct search, in an attempt to provide algorithms that are invariant of starting point or choice of random seed to solve a subfamily of unconstrained and constrained nonlinear multimodal problems. First, we introduce the Adaptive Deterministic Variable Matrix (A-DVM), a deterministic decision variable technique for the Artificial Bee Colony that relies on the measure of spread of the solution set throughout the solution space. We also propose novel procedures to be integrated to the Nelder-Mead in the form of polling steps using approximations of the true gradient. Finally, we devise an adapted augmented lagrangian penalty method to be used together with additional rules to the Nelder-Mead and the A-DVM based Artificial Bee colony to solve nonlinear multimodal constrained problem with nonlinear constraints. Research questions posed in this work are answered by numerical experiments comparing the approaches against standard methods used to solve each problem as well as the state of the art present in the literature.

Contents

List of Figures

List of Tables

Nomenclature

δ_f	vector of objective function differences
$\mathcal{N}(\cdot)$	Neighborhood of a point in $\mathbb{R}^n$
$\hat{\boldsymbol{X}}$	Set of stagnated solutions
$\boldsymbol{I}$	Identity matrix of appropriate dimension
$\boldsymbol{X}$	Solution set of appropriate dimension and cardinality
$\boldsymbol{Y}$	Solution set of the Nelder-Mead representing a simplex
$\nabla_s f$	Simplex gradient in respect to f
$\sigma^+(\boldsymbol{Y})$	Oriented length of simplex
$\text{diam}(\boldsymbol{Y})$	Simplex diameter
$\text{vol}(\boldsymbol{Y})$	Volume of simplex
$\text{von}(\boldsymbol{Y})$	Normalized volume of simplex
$\boldsymbol{e}$	all-ones vector of appropriate dimension
$\boldsymbol{l}$	Vector of lower bounds
$\boldsymbol{u}$	Vector of upper bounds
$\boldsymbol{x}$	solution vector
$L(\boldsymbol{Y})$	matrix of edge lengths
$U(\cdot,\cdot)$	Uniform distributed variable between bounds

Chapter 1

Introduction

Nonlinear optimization problems with deceptive local optima are commonplace in mathematical models that represent real life problem. Most of the times, the closer a model is to the real problem, the higher the probability of the objective function to be multimodal [1]. Derivative-free methods are often used as the first option to tackle particularly difficult formulations because they can provide satisfactory results without the need of thorough understanding of the problem in order to devise a relaxation/modification to the mathematical model [2]. This can be attributed to the fact that Derivative-free algorithms do not rely on information from the curvature of the function such as the first or second derivatives.

Two subfamilies of the derivative-free methods, the direct search and Swarm Intelligence family of algorithms, are used in a vast range of nonlinear problems, constrained or unconstrained as a last resort to solve problems that are so ill-conditioned that nothing else is possible. This is possible because they do not construct linear or quadratic approximation models, guiding incumbent solutions solely by their objective function value. This flexibility comes at a heavy price though, they are heavily dependent on pure randomness, such as the choice of starting point, update process of solution set and even when choosing components of a solution vector. In sum, randomness compromise their robustness [2].

In the interest of narrowing down the scope of the derivative-free algorithms solely to nonlinear ill-conditioned multimodal problems, a natural course of action would be to compromise their flexibility in favor of invariance. For this to be possible, we integrate procedures that are based on deterministic processes, where it attempts to exploit the nature of the problem to better guide the solutions. By doing so, not only we improve the reliability of such algorithms, but also enhance their interpretability, so that it may be possible to understand their internal processes.

In this thesis, we specifically study constrained and unconstrained multimodal optimization problems with less than 100 dimensions. These problems can be encountered in fields such as civil engineering, financial engineering, chemistry and electric engineering. These problems not only are nondifferentiable and highly nonconvex, but their constrained versions also feature nonlinear constraints [1]. Because derivative-free algorithms are the only viable option to solve these problems, we focus on two of them in particular, the Artificial Bee Colony (ABC) and the Nelder-Mead, that employ deterministic procedures in order to improve their robustness and overall performance.

A common weakness of metaheuristics is their extreme dependence on probability and lack of information of the problem space. Even if there is a global optimal solution near the current search point, they might run the risk of overlooking it. As a means to avoid it, we introduce an exhaustive search mechanism, named A-DVM, that can be included to any populational heuristic. In this thesis we limit to study its effects into the Artificial Bee Colony algorithm. Meanwhile, the Nelder-Mead algorithm does not depend on probability,but has no mechanism for global search. In consideration of the multimodality of the target problem, we make it a kind of metaheuristic algorithm by restarting from a stochastically selected initial simplex after convergence.

More specifically, we propose a selection method of decision variables for the ABC that exploits the diagonal argument of Cantor in order to balance the emphasis on local and global search of the algorithm. Then, we develop a reinforced Nelder-Mead that employs a polling step, a deterministic procedure from model-based derivative free algorithms, based on an approximation of the simplex gradient. Lastly, we also introduce a penalty method for the two algorithms to handle constrained multimodal problems with nonlinear constraints, together with the new rules for decision variable selection and restart of the solution set that exploits the nature of these problems.

Numerical comparisons are presented to corroborate all research questions posed in this thesis.

1.1 General Objective

This thesis' objective is to present novel derivative-free algorithms that employ methods that are rooted in deterministic procedures as an attempt to reduce their reliance on pure randomness focusing on solving small scale nonlinear multimodal problems, be it constrained or unconstrained. The performance of the new techniques and the research questions related to the methods are assessed by means of several numerical experiments involving distinct families of multimodal instances.

1.1.1 Specific Objectives

The specific objectives are described as follows:

1. Introduce a deterministic decision variable selection process to derivative-free methods, in particular the Artificial Bee Colony algorithm. Investigate potential improvements to the original and variants to solve a restricted family of multimodal problems.

2. Introduce a novel Nelder-Mead direct search algorithm which uses a procedure that is common in model-based algorithms, the construction of the polling step. The polling then is used as a way of monitoring the geometry of the solution set and restarting it in another region of the search space, if applicable.

3. Propose an adapted penalty method for derivative-free algorithms that uses a set of solution in their optimization process and investigate whether this method brings an improvement in the performance of the aforementioned methods.

1.2 Thesis Organization

The thesis is divided into four main chapters:

1. **Theoretical Background**. This chapter provides the theoretical background essential to understanding of the following three chapters. Detailed information on the Artificial Bee Colony and the Nelder-Mead are provided. Furthermore, the complexity of the algorithms as well as some information on some variants are also given.

2. **Adaptive Decision Variable Matrix**. We propose a method to select decision variables to be integrated to the Artificial Bee Colony, a population heuristic from the family of Swarm Intelligence algorithms. This method construct a binary selection matrix to choose components of solution vectors to undergo update steps. The selection matrix is the result of the composition of two other matrices that choose decision variables by a random and deterministic procedure, respectively. The degree of how much the deterministic selection is favored over the random selection is regulated by a self adaptive parameter obtained from the estimation of the spread of the solutions in the search space. The A-DVM is tested on multimodal unconstrained minimization problems to assess its robustness compared to other derivative-free methods.

3. **Simplex gradients to the Nelder-Mead**. We introduce a novel version of the Nelder-Mead that employs a polling step to restart or merge the solution set with another solution set. We do so as an attempt to introduce invariance of starting point to the algorithm, so it can be able to properly converge to local optima in highly multimodal instances. In the polling step, a polling set is constructed by means of a line search using a descent direction obtained from the simplex gradient, an approximation of the true gradient. To ensure that the simplex gradient is a well-enough approximation of the true gradient, the solution set is constructed from a positive spanning basis. Efficacy of the new algorithm is assessed by a large benchmark suite that features nonlinear multimodal problems.

4. **Deterministic procedures to Derivative-free algorithms for constrained nonlinear multimodal problems**. Procedures for the algorithms explained in the previous chapters are proposed to solve multimodal nonlinear constrained problems with nonlinear constraints. In this chapter our contribution is threefold. First, an adapted augmented lagrangian penalty method for solution sets is introduced. Second, an extension of the A-DVM to use extra rules for constrained problems. Third, an initialization method and safeguards for the Nelder-Mead to solve constrained instances. The new algorithms are assessed by means of eight engineering design problems.

Chapter 2

Theoretical Background

This chapter introduces a collection of fundamental concepts essential to the understanding of the upcoming chapters. The two derivative-free methods which were extensively studied in this work are detailed in full.

From now on, we assume that any optimization algorithm detailed in this work solve an optimization problem of the following form

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && l_j \le x_j \le u_j, \quad j = 1, \ldots, n. \end{aligned} \tag{2.1}$$

Where $f : \mathbb{R}^n \to \mathbb{R}$ is nonlinear, multimodal and can be nondifferentiable or/and noncontinuous. We assume each variable is box-constrained, otherwise $l_j = -\infty$ and $u_j = \infty$.

2.1 Artificial Bee Colony

Artificial Bee Colony (ABC) is a Swarm Intelligence (SI) heuristic developed by Karaboga [3] based on the mathematical model of the foraging and information sharing behavior of honey bees to solve unconstrained optimization problem in the form of (2.1). The popularity of the ABC is due to its simple design and easiness to adapt to other families of optimization problems [4]. The canonical ABC described in Algorithm 1 is composed of four main steps, initialization, employed bees, onlooker bees, and scout bees step. In the initialization step, the solution set $\boldsymbol{X}$ is initialized based on specific rules. Then, solutions are sampled and updated by local and global search procedures iteratively until a stopping criterion is met.

ABC has three tunable parameters, the solution set size SN; the maximum number of iterations MCN; and the solution stagnation threshold Lit. A brief description of each step is given below. Let $\boldsymbol{X} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{SN}\}$ be the solution set where each $\boldsymbol{x}^i$ is a point in $\mathbb{R}^n$.

2.1.1 Initialization

If no information of the solution space is provided, $\boldsymbol{x}^i$ is sampled from a uniform distribution in the feasible interval $[l_j, u_j]$ of each decision variable x_j^i, i.e., for $i = 1, 2, \ldots, SN$,

Algorithm 1: Canonical Artificial Bee Colony

```
  Input: $f, MCN, SN, Lit$
1 $\boldsymbol{X} \leftarrow$ Initialize($SN$)
2 for $t \leftarrow 1$ to $MCN$ or until stopping criteria do
3     $\boldsymbol{X} \leftarrow$ EmployedBeesCycle($\boldsymbol{X}, f$)
4     $\boldsymbol{X} \leftarrow$ OnlookerdBeesCycle($\boldsymbol{X}, f$)
5     $\boldsymbol{X} \leftarrow$ ScoutBeesCycle($\boldsymbol{X}, f, Lit$)
6 end
7 return $\boldsymbol{X}$
```

$$\boldsymbol{x^i} = u_j + \mathrm{U}(0,1)\left(l_j - u_j\right), j = 1, \dots n. \tag{2.2}$$

where $\mathrm{U}(0,1)$ denotes a uniform distribution between 0 and 1. A counter $l_{c_i} = 0$ to indicate unsuccessful updates is initialized for each i.

2.1.2 Employed bees cycle

A randomly chosen component x^i_j of each solution $\boldsymbol{x^i} \in \boldsymbol{X}$ is moved by a random step size towards the jth component of some $\boldsymbol{x}^k \in \boldsymbol{X}, k \neq i$. Therefore $\boldsymbol{x}^i$ is updated into:

$$x^i_q = \begin{cases} x^i_j + \phi\left(x^i_j - x^k_j\right) & \text{if } q = j, \\ x^{i'}_q, & \text{otherwise,} \end{cases} \tag{2.3}$$

where $\phi \in \mathrm{U}(-1,1)$. To verify if the update step was successful, the value of f is evaluated and a greedy selection is done for each $i = 1, 2 \dots, SN$:

$$\boldsymbol{x^i} = \begin{cases} \boldsymbol{x}^i + (x^{i'}_j - x^i_j)\boldsymbol{e}_j,\ l_{c_i} = 0 & \text{if } f(\boldsymbol{x}^i + (x^{i'}_j - x^i_j)\boldsymbol{e}_j) \leq f(\boldsymbol{x}^i), \\ \boldsymbol{x}^i,\ l_{c_i} = l_{c_i} + 1 & \text{otherwise,} \end{cases}. \tag{2.4}$$

where $\boldsymbol{e}_j$ is the jth fundamental vector. Needless to say, if (2.3) fails, then (2.4) will flag (2.3) as a failed update.

2.1.3 Onlooker bees phase

The solution $\boldsymbol{x}^i \in \boldsymbol{X}$ is chosen with probability p_i according to a weighted roulette selection scheme and updated using (2.3). This step can be thought of enhancing local search for solutions with better objective function values. The probability p_i is determined for each solution $\boldsymbol{x}^i \in \boldsymbol{X}$ as follows,

$$p_i = \frac{F(\boldsymbol{x}^i)}{\sum_{i=1}^{SN} F(\boldsymbol{x}^i)}, \tag{2.5}$$

where $F(\cdot)$ is the adjusted objective function value:

$$F(\boldsymbol{x}^i) = \begin{cases} \frac{1}{1+f(\boldsymbol{x}^i)} & \text{if } f(\boldsymbol{x}^i) \geq 0 \\ \left|1 + f(\boldsymbol{x}^i)\right| & \text{otherwise.} \end{cases} \tag{2.6}$$

2.1.4 Scout bees phase

Let $\tilde{\boldsymbol{X}} = \{\boldsymbol{x}^k \in \boldsymbol{X} \mid l_{c_k} \geq Lit\}$ denote the set of solutions flagged as stagnated. A new point in $\mathbb{R}^n$ is resampled using (2.2) if $\tilde{\boldsymbol{X}} \neq \emptyset$. This prevents the algorithm from premature convergence to bad local optima and increases the number of explorations. The parameter Lit is commonly defined equal to $SN \cdot n$. If $\tilde{\boldsymbol{X}} \neq \emptyset$, then $\boldsymbol{x}^w \in \text{argmax}\{f(\boldsymbol{x}) \mid \boldsymbol{x} \in \tilde{\boldsymbol{X}}\}$ is always chosen to be resampled.

2.1.5 ABC Variants

Due to the modular nature of the ABC it is easy to make changes to any of the three steps of the algorithm [5]. Many modifications were proposed by several researchers to improve on some difficulties of the algorithm such as, inclusion of memory to assist local search; efficient mechanisms to displace solutions stuck in local optima; handle high dimensional ($n > 100$) problem instances; changes to the update rule; and initialization of solutions using local information. As observed by Aydin et al. [6], the ABC variants often differ in some core components such as, the initialization of the first solution set; the update step in the employed and onlooker bees; the computation of selection probabilities in the onlooker bees, the way of displacing solutions in the scout step.

Some well-known and successful variants include the chaotic ABC version of Alatas and Bilal [7] which uses chaotic maps for solution initialization, the ABC of Akay and Karaboga [8] and Gao and Liu [9] which update multiple decision variables in a single update step. Additionally, the ABC with modified selection scheme based on neighborhood distances by Diwold et al. [10], integration of the Differential Evolution algorithm with the ABC by Xiang et al. [11] and Akay et al. [12] are well known. The reader is encouraged to read the survey of Karaboga et al. [5], Sharma and Bhambu [13] for further information.

2.1.6 Complexity of the Arficial Bee Colony

Complexity of the ABC is measured according to the number of function evaluation calls performed at each iteration. For each iteration the number of FE's is as follows: SN FE's in the employed bees phase; SN or $\frac{SN}{2}$ in the onlooker bees phase depending on the implementation of the ABC; lastly 1 to 0 or SN to 0 in the scout bees phase according to the implementation. The ABC of this work use the former for both onlooker and scout bees phase.

Since there is no need to order the solution set $\boldsymbol{X}$ in the original implementation, the onlooker and employed bees phase is carried out in $\mathcal{O}(n)$ time. Furthermore, the implementation of the scout bees step in this work chooses at most 1 solution, therefore it is also performed in $\mathcal{O}(n)$ time.

2.2 Nelder-Mead Simplex Method

The Nelder-Mead (NM) method is a direct search method developed by Nelder and Mead [14] in 1965 as a modification of the simplex-based direct search method from Spendley, Hext and Himsworth [15] to solve problems of the form of (2.1). The algorithm is regarded as one of the most widely cited of any direct search method, cited in a great number of works from a vast range of fields of study [16]. Several reasons as to why this algorithm is so popular may be due to its easy implementation; the capability to adapt to the local landscape of the objective function; and the low memory requirement. As a common aspect to any direct search method, its efficiency deteriorates the higher the number of dimensions of the problem, a concept reinforced in some works that showed that the NM is able to produce good results to functions that features no more than 10 decision variables [16, 17, 18].

A solution set of the NM consists of a $n \times n+1$ matrix $\boldsymbol{Y} = \left[\boldsymbol{y}^0 \; \boldsymbol{y}^1 \; \cdots \boldsymbol{y}^n \right]$ that describes a simplex, a geometric figure in n dimensions of nonzero volume (also called nondegenerate) that is the convex hull of $n+1$ vertices. Several heuristics to construct $\boldsymbol{Y}$ are available in the literature which will be addressed in full detail in then next section.

At every iteration, each step of the algorithm attempts a geometric transformation to so that the worst point $\boldsymbol{y}^n \in \operatorname{argmax}\{f(\boldsymbol{y}) \mid \boldsymbol{y} \in \boldsymbol{Y}\}$ can be replaced by a better point in the following way:

$$\boldsymbol{y}' = \boldsymbol{c} + \alpha(\boldsymbol{c} - \boldsymbol{y}^n). \tag{2.7}$$

Where $\boldsymbol{c}$ is the centroid of the best n points, $\boldsymbol{y}^n$ is the worst point of the simplex and α is the coefficient associated to the transformation. The steps are: reflection; expansion; internal contraction and external contraction. If all fails, the shrink step takes place, where the entire simplex $\boldsymbol{Y}$ is shrunk from the best vertex of simplex $\boldsymbol{y}^0 \in \operatorname{argmin}\{f(\boldsymbol{y}) \mid \boldsymbol{y} \in \boldsymbol{Y}\}$. At the termination, the algorithm outputs the vertex of the simplex which yielded the least objective function value. Recent versions of the NM differ much from the original proposed in 1965. We describe the most consolidated version written in the work of Lagarias et al. [19] and Wright [20] and implemented as the function **fminsearch** in MATLAB [21]. Algorithm 2 describes the steps of the NM following the guidelines of Conn, Scheinberg and Vicente [2].

Algorithm 2: Nelder-Mead algorithm

```
   Input: $f(\cdot)$, $\boldsymbol{x}_0$, $t$, $\gamma^s$, $\delta^{ic}$, $\delta^{oc}$, $\delta^r$, $\delta^e$
   Output: $\boldsymbol{Y}_t = \{\boldsymbol{y}^0, \boldsymbol{y}^1, \cdots, \boldsymbol{y}^n\}$ where $\boldsymbol{y}^0 = \operatorname{argmin} f(\boldsymbol{y}),\ \boldsymbol{y} \in \boldsymbol{Y}_t$
   Initialization: Initialize simplex $\boldsymbol{Y}_0 = \{\boldsymbol{y}^0, \boldsymbol{y}^1, \cdots, \boldsymbol{y}^n\}$ using $\boldsymbol{x}_0$
1  for $i \leftarrow 1$ to $t$ do
2      if $\boldsymbol{Y}_i$ fails termination test then
3          return $\boldsymbol{Y}_i$
4      $\boldsymbol{Y}_i \leftarrow \boldsymbol{Y}_{i-1}$
5      $\boldsymbol{Y}_i \leftarrow$ Order $(\boldsymbol{Y}_i)$
6      $\boldsymbol{c} \leftarrow$ ComputeCentroid $(\boldsymbol{Y}_i)$
7      $\boldsymbol{y}^r \leftarrow$ Reflection $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^r)$
8      Expansion $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^e)$
9      OutsideContraction $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^{oc})$
10     InsideContraction $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^{ic})$
11     Shrink $(\boldsymbol{Y}_i, \gamma^e)$
12 end
13 return $\boldsymbol{Y}_t$
```

The standard parameter values that control the simplex transformations of the coefficients of each step are $\gamma^s = \frac{1}{2}$, $\delta^{ic} = -\frac{1}{2}$, $\delta^{oc} = \frac{1}{2}$, $\delta^r = 1$, $\delta^e = 2$, note that other notations for them are used in [22, 19]. The parameters must satisfy the following constraints, $\delta^r > 0$, $\delta^e > 1$, $\delta^e >$

$\delta^r,\ 0 < \delta^{oc} < 1,\ 0 < \gamma^e < 1$. A different choice was proposed by Parkinson and Hutchinson [23], a parameter tuning investigation was conducted by Fang and Zahara [24] and Wang and Shoup [25]. Lastly, Gao and Han [26] proposed adaptive parameters for multimodal functions and proved its efficacy.

Each step of the algorithm is explained point-by-point.

1. **Initialization:** A $n \times n + 1$ matrix $\boldsymbol{Y}$ that represents a simplex is constructed from an initial point $\boldsymbol{x}^0 \in \mathbb{R}^n$. Methods invariant of an initial point $\boldsymbol{x}^0$ are unlikely to be good since the performance of the NM hinges on a good enough initial point provided at the initialization.

2. **Order:** Arrange the $n + 1$ columns of $\boldsymbol{Y}$ in an increasing order according to their objective function value. In case of any ties, Lagarias et al.[19] suggests to use the least subscript rule as a tie-breaker.

3. **Compute the centroid:** Compute the centroid of the n best vertices of the simplex,

$$\boldsymbol{c} = \frac{1}{n-1} \sum_{i=0}^{n-1} \boldsymbol{y}^i. \tag{2.8}$$

4. **Reflection:** A point $\boldsymbol{y}^r$ that is the located in the opposite side of the line segment separating $\boldsymbol{c}$ and $\boldsymbol{y}^0$ is computed from (2.7) setting $\alpha = \delta^r = 1$. If $f(\boldsymbol{y}^0) \leq f(\boldsymbol{y}^r) < f(\boldsymbol{y}^{n-1})$, then $\boldsymbol{y}^n$ is replaced by $\boldsymbol{y}^r$ and the iteration is terminated with the new simplex as $\boldsymbol{Y}^{i+1} = \{\boldsymbol{y}^0,\ \boldsymbol{y}^1,\ \ldots,\ \boldsymbol{y}^{n-1},\ \boldsymbol{y}^r\}$.

5. **Expansion:** If $f(\boldsymbol{y}^r) < f_0$, expansion is attempted beyond $\boldsymbol{y}^r$ from $\boldsymbol{c}$ to search for a promising regions. The expanded point is calculated from (2.7) setting $\alpha = \delta^e = 2$. Evaluate $f(\boldsymbol{y}^e)$. If $f(\boldsymbol{y^e}) \geq f(\boldsymbol{y^r})$, replace $\boldsymbol{y^n}$ by $\boldsymbol{y^e}$ and terminate the iteration. Otherwise replace $\boldsymbol{y^n}$ by $\boldsymbol{y^r}$ and terminate the iteration with the new simplex as $\boldsymbol{Y}^{i+1} = \{\boldsymbol{y}^0,\ \boldsymbol{y}^1,\ \ldots,\ \boldsymbol{y}^{n-1},\ \boldsymbol{y}^e\}$.

6. **Contraction:** If $f(\boldsymbol{y}^r) \geq f(\boldsymbol{y}^{n-1})$, a contraction is performed between $\boldsymbol{y^r}$ and $\boldsymbol{y^n}$. Depending on the value of $f(\boldsymbol{y^r})$, either a outside or inside contraction can take place. If either fail, a shrink is done.

   - **Outside contraction:** If $f(\boldsymbol{y^r}) < f(\boldsymbol{y^n})$, an outside contraction is performed, point $\boldsymbol{y^{oc}}$ is computed from (2.7) setting $\alpha = \delta^{oc} = 0.5$ and $f(\boldsymbol{y^{oc}})$ is evaluated. If $f(\boldsymbol{y^{oc}}) \leq f(\boldsymbol{y^r})$, replace $\boldsymbol{y^n}$ by $\boldsymbol{y^{oc}}$ and terminate iteration with the new simplex as $\boldsymbol{Y}^{i+1} = \{\boldsymbol{y}^0,\ \boldsymbol{y}^1,\ \ldots,\ y^{n-1},\ \boldsymbol{y^{oc}}\}$.
   - **Inside contraction:** If $f(\boldsymbol{y}^r) \geq f(\boldsymbol{y}^n)$, perform an inside contraction, calculate $\boldsymbol{y^{ic}}$ from (2.7) setting $\alpha = \delta^{ic} = -0.5$ and evaluate $f(\boldsymbol{y^{ic}})$. If $f(\boldsymbol{y}^{ic}) \leq f(y^r)$, replace $\boldsymbol{y^n}$ by $\boldsymbol{y^{ic}}$ and terminate iteration with the new simplex as $\boldsymbol{Y}^{i+1} = \{\boldsymbol{y}^0,\ \boldsymbol{y}^1,\ \ldots,\ \boldsymbol{y}^{n-1},\ \boldsymbol{y^{ic}}\}$.

7. **Shrink:** If everything else fails, evaluate f at the n points $\boldsymbol{y}^0 + \gamma^s(\boldsymbol{y}^i - \boldsymbol{y}^0), i = 1, \ldots, n$, and terminate the iteration with the new simplex $\boldsymbol{Y}^{i+1} = \{\boldsymbol{y}^0,\ \boldsymbol{y}^0 + \gamma^s(\boldsymbol{y}^i - \boldsymbol{y}^0)\ i = 1, \ldots, n\}$. A standard choice for γ^s is $\frac{1}{2}$.

8. **Termination test:** At the beginning of the iteration the simplex undergoes several termination tests that verifies many conditions regarding the structure of the simplex. Usually these tests verifies whether the simplex has converged to a point, if it became too small or has collapsed into a subspace. Singer and Singer [27] and Nazareth and Tseng [28] suggest monitoring the volume of the simplex as a efficient termination test. Depending on the implementation and purpose of the NM, the simplex may be restarted instead of terminated as an attempt to search for other promising regions from another starting point [29, 30].

Measuring the complexity of the NM in relation to the number of function evaluations (FE) per iteration is simple. For each iteration the number of FE's is as follows: 1 if the iteration ends in a reflection; 2 if the iteration ends in an expansion or contraction; and $n + 2$ if the iteration ends in a shrink. Ordering of vertices is done in $\mathcal{O}(n)$ time using an insertion sort, and the centroid $\boldsymbol{c}$ can be calculated as a rank-1 update from the old centroid in $\mathcal{O}(n)$ time. For more details, we suggest the works of Singer and Singer [31] and Smith [32].

We first introduce some metrics that monitor the behavior of a simplex, then definitions of positive bases and well-poisedness for linear interpolation models. The measures necessary for the understading are the diameter (diam), oriented length (σ^+), the volume (vol) and the normalized volume (von). We begin with the definition of diameter:

$$\text{diam}(\boldsymbol{Y}) = \max_{0\leq i<j\leq n} \left\|\boldsymbol{y}^i - \boldsymbol{y}^j\right\|, \tag{2.9}$$

A less expensive operation called oriented length, can be used instead of the diameter:

$$\sigma^+(\boldsymbol{Y}) = \max_{1\leq i\leq n} \left\|\boldsymbol{y}^i - \boldsymbol{y}^0\right\|. \tag{2.10}$$

Where it can be easily seen that $\sigma^+(\boldsymbol{Y}) \leq \text{diam}(\boldsymbol{Y}) \leq 2\sigma^+(\boldsymbol{Y})$. To distinguish simplices that have the same shape, e.g., $\boldsymbol{Y}$ and $c\boldsymbol{Y},\ c > 0$, the volume of the simplex is an indispensable measure:

$$\text{vol}(\boldsymbol{Y}) = \frac{|\text{det}(L(\boldsymbol{Y}))|}{n!}, \tag{2.11}$$

where $L(\boldsymbol{Y})$ is the matrix of edge lengths defined as:

$$\text{L}(\boldsymbol{Y}) = \begin{bmatrix}\boldsymbol{y}^1 - \boldsymbol{y}^0 & \boldsymbol{y}^2 - \boldsymbol{y}^0 & \dots & \boldsymbol{y}^{n-1} - \boldsymbol{y}^0 & \boldsymbol{y}^n - \boldsymbol{y}^0\end{bmatrix}^\top, \tag{2.12}$$

assuming that $\boldsymbol{Y}$ is ordered by the objective function value. Since a simplex forms an affinely independent set, vol$(\boldsymbol{Y}) > 0$ invariant of the choice of centering point of the simplex [2]. However the volume is not scale invariant so the normalized volume von is customarily used.

$$\text{von}(\boldsymbol{Y}) = \text{vol}\left(\frac{1}{\text{diam}(\boldsymbol{Y})}\boldsymbol{Y}\right) = \frac{|\text{det}(L(\boldsymbol{Y}))|}{n!\text{diam}(\boldsymbol{Y})^n}. \tag{2.13}$$

To cut down the computing power spent in the calculation of von(Y), Tseng [33] disregards $n!$ in (2.13). Lastly, on the subject of the volume of the simplex, we introduce the following lemma that was proved by Lagarias et al. [19] to explain the changes in the volume of a NM simplex throughout the iterations.

Lemma 2.2.1. Volume and nondegeneracy of Nelder-Mead simplices.

1. If the initial simplex $\boldsymbol{Y}$ is nondegenerate, so are all subsequent Nelder-Mead simplices.
2. Following a nonshrink of type τ, $\text{vol}(\boldsymbol{Y}_{k+1}) = |\tau|\text{vol}(\boldsymbol{Y}_k)$.
3. Following a shrink step at iteration k, $\text{vol}(\boldsymbol{Y}_{k+1}) = \sigma^n\text{vol}(\boldsymbol{Y}_k)$.

2.2.1 Convergence of the Nelder-Mead

We provide a brief background of the prominent works which studied convergence analysis and developed of lower bounds of the Nelder-Mead in order to support our propositions in the sections to follow.

For better legibility, we use the following notations regarding the search space of f in accordance to Audet and Warren[34]. $f \in \mathcal{C}^0$ means that f is nondifferentiable, while $f \in \mathcal{C}^1$ and $f \in \mathcal{C}^2$ means that f is once and twice differentiable, respectively, i.e., gradient ∇f and Hessian H can be computed. $f \in \mathcal{C}^+$ signifies that f is Locally Lipschitz continuous at all $x \in \mathbb{R}^n$ with Lipschitz constant K, so for example, $f \in \mathcal{C}^{1+}$ is differentiable and the gradient ∇f is Lipschitz continuous with Lipschitz constant K.

The Nelder-Mead has been shown to not converge on some instances of $f \in \mathcal{C}^1$ functions [22] that are strictly convex, contradicting the assumption that it was globally convergent if the function was convex. Using a particular starting point $\boldsymbol{x}_0$ from the well-known McKinnon function, the authors have shown that after a set amount of iterations, the NM enters in a state where it only does internal contractions and reflections(RFIC). Lagarias et al. [19, 35] has built upon [22] conclusions and presented convergence properties to a minimizer for 1 for the standard NM and to 2 dimension using a version that does not perform the expansion step. Kelley [36] proposes an oriented restart method to replace the shrink operation, providing an sufficient descent condition based on the simplex gradient. From those, the following partial convergence result is established: if the Nelder-Mead algorithm satisfies a sufficient descent condition at all but a finite number of iterations, and if no shrink steps are performed, then the accumulation points of the sequence of iterates of the simplex are stationary points of f.

Nondegeneracy of the volume of the simplex and proofs that the sequence of iterates are bounded away from 0 can be found in [30, 37, 38]. The last two propose convergent versions of the NM that relies on meshes and frames spanned by positive bases to guarantee convergence in instances where $f \in \mathcal{C}^1$.

The following theorem summarizes the properties of the NM, based on the observations of Conn et. al [2].

Theorem 2.2.1. Consider the Nelder-Mead algorithm (Algorithm 2) to a function f bounded from below on $\mathbb{R}^n$, starting with a nondegenerate simplex $\mathit{\Delta}_0$.

1. The sequence $\{y^{k^0}\}$ is convergent
2. If only a finite number of shrinks occur, then all of the $n + 1$ sequences $\{y^{k^i}\}, i = 0, \ldots, n$, converge and their limits $y^{*^i} (i = 0, \ldots, n)$ satisfy $y^{*^0} \leq y^{*^1} \leq \cdots \leq y^{*^n}$.
3. If only a finite number of operations that are not shrinks were to occur, then all vertices of the simplex converge to a single point.

Proof. First and second assertions are based on the fact that monotonically decreasing sequences bounded from below converges [2]. For third assertion, if no shrinks occur, then f is strictly convex. If f is strictly convex, then the Nelder-mead algorithm is sure to converge to an accumulation point, as proven by Lagarias [19] and Conn et al. [2]. $\square$

2.2.2 Simplex Initialization

It is important to remember is that it is a local search global optimization algorithm, i.e., it forfeits a search of a provable global optimum in favor of a "good enough" local optimum $\boldsymbol{x}^*$ solution located in a neighborhood $\mathcal{N}$ where $\mathcal{N}(\boldsymbol{x}^*) = \{\boldsymbol{x} \in \mathbb{R}^n : \|\boldsymbol{x} - \boldsymbol{x}^*\|_2 \leq \epsilon\}$ [39]. Therefore, the performance and robustness of the NM relies strongly on the chosen starting point, size and structure of the starting simplex [40]. Heuristics to build an initial simplex $\boldsymbol{Y}_0$ are vast and obscure. Although the definition of a simplex says that it is an affinely independent set that spans the convex hull of a convex set [41], in NM the only requirement of $\boldsymbol{Y}_0$ is to be non-degenerate so that the volume can be properly monitored and its convergence properties be maintained. Most heuristics do not provide a sound mathematical rigor that explain why it works, save a few that relies on the construction of positive bases [2, 42].

We discuss two simplex initialization methods: the most well-known heuristic to date, seen in works as old as 1985 like [43], mentioned in the book written by Haftka [44] and the standard initializer in the *fminsearch* function in MATLAB [21] and the SciPy package [45]; and a recent initialization method by Bolduc et al. [42] which proposed an uniform simplex with arbitrary orientation.

Standard Simplex Initialization

This heuristic is mentioned in many works in the literature and is the initialization heuristic implemented in MATLAB and Scipy. A $n+1$ simplex $\boldsymbol{Y}_0 = [\boldsymbol{y}^0\, \boldsymbol{y}^1\, \cdots\, \boldsymbol{y}^n]$ is constructed from an initial point $\boldsymbol{x}^0 \in \mathbb{R}^n$ by the following rule:

$$\boldsymbol{y}^i = \begin{cases} (x_1^0,\ \ldots,\ x_{i-2}^0,\ c_P x_{i-1}^0,\ x_i^0,\ \ldots,\ x_n^0)^\top & \text{if } x_{i-1}^0 \neq 0 \\ (x_1^0,\ \ldots,\ x_{i-2}^0,\ 0.00025,\ x_i^0,\ \ldots,\ x_n^0)^\top & \text{otherwise.} \end{cases} \tag{2.14}$$

Where c_P is set to $1 + 0.05$. It can be clearly observed that the oriented length σ^+ of the simplex is rather small and the simplex in not coordinate invariant, which could be troublesome if either the lower bound l_i or upper bound u_i of the i-th box constraint is above or below 0.00025.

Uniform Simplex

The term "uniform simplex" has first appeared in the book of Audet and Hare, "Derivative-free and Blackbox Optimization" [34] as an exercise left to the reader to derive a simplex centered in the origin. After its first appearance, Bolduc et al. [42] formalized the Uniform Simplex, developed bounds and properties and extended to be able to be rotated towards any orientation. Let $\boldsymbol{c}$ be the centroid of the simplex $\boldsymbol{c} = \frac{1}{n+1}\sum_{i=0}^{n} \boldsymbol{x}_i$, we refer to $\mathbb{U}$ to what the authors called a "canonical uniform simplex" as a simplex that has the following properties:

(i) $\mathbb{U}$ is nondegenerate, i.e., vol$(\mathbb{U}) > 0$.

(ii) $\|\boldsymbol{x}_i - \boldsymbol{c}\| = m > 0$ for some $m \in \mathbb{R}$ and $i = 0, 1, \ldots, n$. For this case $m = 1$.

(iii) $(\boldsymbol{x}_i - \boldsymbol{c})^\top(\boldsymbol{x}_j - \boldsymbol{c}) = k$ for some $k \in m \in \mathbb{R}$ for $i, j = 0, 1, \ldots, n, i \neq j$.

(iv) $\boldsymbol{c} = 0$.

(v) The subset of $\boldsymbol{M} = [\boldsymbol{y}^1, \ldots, \boldsymbol{y}^n]$ of $\mathbb{U}$ is upper triangular.

(vi) All entries of diag$(\boldsymbol{M})$ are positive.

We argue that uniform simplex $\mathbb{U}$ is a regular simplex on $\mathbb{R}^n$ centered at 0 because it satisfies properties (i), (ii) and (iii) [46]. Construction of the canonical uniform simplex $\mathbb{U}$ can be done analytically. Any simplex $\mathbb{U}$ is defined by the following matrix form:

$$\begin{aligned}[\boldsymbol{M}|\boldsymbol{y}^0] &= [\boldsymbol{y}^1\ \boldsymbol{y}^2\ \boldsymbol{y}^3\ \cdots\ \boldsymbol{y}^n|\boldsymbol{y}^0] \\ &= \begin{bmatrix} a_1 & -\frac{a_1}{n} & -\frac{a_1}{n} & -\frac{a_1}{n} & \cdots & -\frac{a_1}{n} & -\frac{a_1}{n} \\ 0 & a_2 & -\frac{a_2}{n-1} & -\frac{a_2}{n-1} & \cdots & -\frac{a_2}{n-1} & -\frac{a_2}{n-1} \\ 0 & 0 & a_3 & -\frac{a_3}{n-2} & \cdots & -\frac{a_3}{n-2} & -\frac{a_3}{n-2} \\ 0 & 0 & 0 & a_4 & \cdots & -\frac{a_4}{n-3} & -\frac{a_4}{n-3} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & a_n & -a_n \end{bmatrix},\end{aligned} \tag{2.15}$$

where the only difference between column $\boldsymbol{y}^0$ and column $\boldsymbol{y}^n$ is that the last component of $\boldsymbol{y}^0$ is $-y^n_n$. where a is defined as follows:

$$a_i = \sqrt{\frac{(n-1+1)(n+1)}{n(n-1+2)}}. \tag{2.16}$$

The most pertinent point of this simplex is any uniform simplex can be made if $\mathbb{U}$ is scaled by a unit vector by a factor of m. Moreover, $\mathbb{U}$ can also be scaled by a $n \times n$ matrix $\boldsymbol{M}$ and still be a uniform simplex centered at the origin. In fact, the authors introduce a rotation of $\mathbb{U}$ towards a general direction $\boldsymbol{d}$ where $\|\boldsymbol{d}\|_2 = 1$ by a Householder rotation. Given identity matrix I and unit vector $\boldsymbol{d}$, a Householder matrix $\boldsymbol{H}$ is defined the following way:

$$\boldsymbol{H} = \boldsymbol{I} - 2\boldsymbol{d}\boldsymbol{d}^\top. \tag{2.17}$$

where the new rotated simplex $\mathbb{U}'$ is calculated as follows:

$$\mathbb{U}' = \boldsymbol{H}\mathbb{U}. \tag{2.18}$$

Clearly, $\mathbb{U}$ is a positive spanning basis, meaning that it is affine independent. Note that for each vertex, the coordinates of each component is within the interval $[-1, 1]$.

Although the uniform simplex has not been proposed specifically as an initialization method for the Nelder-Mead, it is a valid simplex that is equipped with properties that guarantees its non-degeneracy. Therefore, we consider the uniform simplex to be a suitable method to build an initial simplex for the NM.

Chapter 3

A Deterministic Adaptive Decision Variable Matrix Decision Variable Selection Scheme for Swarm Intelligence Algorithms

Artificial Bee Colony (ABC) is a swarm intelligence (SI) heuristic for optimization problems inspired by the foraging behavior of honey bees. Following its conception, improvements to the search capabilities of the original ABC were proposed by many researchers. The great majority of these propositions centered around changes to the initialization of solutions in the solution space, update procedure of the first two phases and selection method in the onlooker phase [6]. Despite differences between each variant, they all share a common trait: the solution update rule chooses one to n decision variables with equal probability under a random uniform distribution. This may allow for a better exploration of the search space and prevent solutions to collapse in the same subspace at later iterations. However, issues to the consistency and convergence of the algorithm may arise due to this design choice.

Besides the uniformity of choice of decision variables, it has been observed that most ABC variants handle poorly problems whose objective functions are multimodal. Adaptations of the ABC to solve problems from this family include changes to the main update equation; adoption of a self-adaptive solution set growth/shrink scheme; and adaptations to the re-sampling step of stagnated solutions [6]. Despite efforts, the ABC still lacks a way to understand how the solutions are fitted and how apart they are in the objective function landscape.

Taking into account the deficiencies observed in the ABC mentioned above, we propose the Adaptive Decision Variable Matrix (A-DVM), a self-adaptive decision variable selection procedure that is an extension of a deterministic solution variable scheme developed in Mollinetti et al. [47]. A-DVM builds an augmented binary matrix that automatically balances deterministic and random decision variable selection to maintain a healthy amount of exploration in the early iterations while emphasizing exploitation in later stages. Levels of exploration and exploitation are monitored by an indicator of how many solutions cover the search space. The chosen estimator is the Δ value, a measure proposed by Morrison [48], which provides a reliable assessment of the shape of the distribution of the solutions along the main and peripheral axes of search. To validate the proposed approach, A-DVM is incorporated into the original ABC and several state-of-the-art variants and evaluated in a test set that features 15 multimodal unconstrained problems. Results are compared to the original counterparts of the ABCs, as well as to some well-established optimization algorithms such as the Particle Swarm Optimization (PSO) and Differential Evolution (DE).

The contributions that the A-DVM brings is twofold. First, a selection scheme that attempts to establish a balance between the global and local search along with the iterations so that the search can be conducted more efficiently. Naturally, it can be incorporated into any version of the ABC without interfering with any other mechanism. Second, a mean to assess how solutions of the solution set are spread throughout the search space and how well they fit the objective function value landscape. Such information is very beneficial in guiding solutions out of deceptive local optima when considering multimodal problems.

This chapter is organized as follows: Section 3.1 discusses the main issues behind the fully randomized selection. Section 3.2 explains the idea behind the A-DVM. Section 3.3 reports the experiments. Lastly, Section 3.4 outlines the conclusion of the chapter and points future directions.

3.1 Issues of Randomization

Population-based optimization methods usually employ randomization. By choosing step sizes, decision variables or even target solutions at random during the update steps, population-based optimization methods can "cover more ground" in the search space effortlessly. This is a key element to the success of population-based heuristics, but not without some unintended side effects.

For the sake of clarity, we refer in accordance to [49] to a neighborhood $\mathcal{N}(\cdot)$ as the classical definition of a Euclidean ball centered at a point $\boldsymbol{x}^k$,

$$\mathcal{N}(\boldsymbol{x}^k) = \{\boldsymbol{x} \in \mathbb{R}^n : \left\|\boldsymbol{x} - \boldsymbol{x}^k\right\|_2 \leq \epsilon\}, \tag{3.1}$$

where $\|\cdot\|_2$ is the ℓ_2 norm and $\epsilon \geq 0$. Assume that a stochastic heuristic, such as the Artificial Bee Colony (ABC), runs infinitely on a problem $(f, S) \in \mathcal{P}$, where f is the objective function, S is the feasible set, and $\mathcal{P}$ is a problem family. Moreover, borrowing some concepts explained in [39], let $X_{f,S}(\omega) = \{f(\omega_k) \mid k = 1, 2, \ldots\}$ denote the infinite sequence of iterates generated by the heuristic where $\omega = \{\omega_k \mid k = 1, 2, \ldots\}$ is a sequence of random numbers distributed independently from (f, S). Let $X'_{f,S}$ and $\bar{X}_{f,S}$ denote the set of accumulation points and the closure of the sequence $X_{f,S}(\omega)$. Lastly, let $X^*_{f,S}$ denote the set of global optima. Clearly no X^* can be "seen" if $\bar{X}_{f,S} \cap X^*_{f,S} = \emptyset$ or $\bar{X}'_{f,S} \cap X^*_{f,S} = \emptyset$.

In the following section, we will see how randomization affects the performance of the ABC.

3.1.1 An Analysis of the ABC decision variable selection

Often overlooked, a common aspect of the ABC variants is that decision variable x^i_j is chosen according to the same uniform distribution with for all $j = 1, 2 \ldots, n$ during the Onlooker and Employed bees steps.

Let $Pr(x^i_j)$ be the probability that x^i_j is chosen in (2.3). For each situation below, we assume that $(f, S) \in \mathcal{P}$, $\bar{X}_{f,S} \bigcap X^*_{f,S} \neq \emptyset$ and $X_{f,S}(\omega)$ is monotonically decreasing, i.e., $f(\omega_1) \geq$

$f(\omega_2) \geq \ldots$. The original ABC chooses a single x^i_j each time it calls (2.3) during the Onlooker and Employed bees step. We need to notice the following issues brought by the process of selecting j randomly.

1. **Failed update steps cause solutions to be trapped in basins of attraction:** Choosing the same wrong decision variable many times fails to move solutions out of basins of attraction, contributing to wasteful iterations, premature convergence and needless flagging of solutions at the scout bees step. Let $\boldsymbol{x}^{w^*} \in X'_{f,S}, \boldsymbol{x}^{k^*} \in X'_{f,S}, \boldsymbol{x}^{s^*} \in X'_{f,S}$. Suppose that $\boldsymbol{x}^w \in \mathcal{N}(\boldsymbol{x}^{w^*})$ and $\boldsymbol{x}^k \in \mathcal{N}(\boldsymbol{x}^{k^*})$. Also suppose $f(\boldsymbol{x}^w) = f(\boldsymbol{x}^{w^*})$, $f(\boldsymbol{x}^w) \approx f(\boldsymbol{x}^{s^*})$, $f(\boldsymbol{x}^w) > f(\boldsymbol{x}^{k^*})$, and $\mathcal{N}(\boldsymbol{x}^{k^*})$ and $\mathcal{N}(\boldsymbol{x}^{s^*})$ are adjacent to $\mathcal{N}(\boldsymbol{x}^{w^*})$.

   Lastly, let x^w_j be a component of $\boldsymbol{x}^w$ such that a successful update moves $\boldsymbol{x}^w$ to $\mathcal{N}(\boldsymbol{x}^{k^*})$ while an update to x^w_q for any $q \neq j$ moves $\boldsymbol{x}^w$ to $\mathcal{N}(\boldsymbol{x}^{s^*})$. $\boldsymbol{x}^w$ is moved in (2.3) one axis at a time, if x^w_j is chosen, then (2.4) accepts $\boldsymbol{x}^{w'}$ and $l_w = 0$. Otherwise, l_w is incremented by 1 every time $\boldsymbol{x}^{w'}$ is rejected by (2.4). If each component is chosen in (2.3) with equal probability, then the probability of x^w_j to be chosen is $Pr(x^w_j) = 1/n$. Therefore, $\boldsymbol{x}^w$ has a probability of $1 - 1/n$ to move to a basin of attraction $\mathcal{N}(\boldsymbol{x}^{s^*})$ similar to $\mathcal{N}(\boldsymbol{x}^{w^*})$, and probability $1/n$ to move to a more promising region $\mathcal{N}(\boldsymbol{x}^{k^*})$.

2. **Decision variables may never be chosen:** If the problem is of high dimensional $(n > 100)$ or the evaluation function $f(\boldsymbol{x})$ is so expensive that only a limited number of objective function calls are allowed, there will be at least a component x^i_j that may never be chosen in (2.3). Let $Pr(\sim x^i_j) = 1 - 1/n$ be the probability of x^i_j *not* to be chosen at (2.3). Then the probability of x^i_j *not* to be chosen be at the end of MCN iterations is $Pr_{MCN}(\sim x^i_j) = \prod_{n=1}^{2MCN}(1 - 1/n)$. It is clear that $Pr_{MCN}(\sim x^i_j)$ converges to 0 as the number of iterations goes to infinity. If $n > 100$ and ABC runs $t \approx n$ iterations, then $Pr_{MCN}(\sim x^i_j) \ll 1$, so x^i_j is not chosen in (2.3).

At first glance, there would be two ways to resolve these issues. Either assign a non-equal probability to choose the decision variable or choose more than one x^i_j at (2.3) to be updated simultaneously. We disprove the effectiveness of these "quick fixes" through following arguments.

- Changing the choice probabilities of decision variables to be unequal would not solve **issue $\#\mathbf{2}$** in high dimensional problems because $Pr_{MCN}(\sim \boldsymbol{x}_j)$ still converges to 0. A sufficient measure, in this case, would to keep previously chosen components in memory. This only increases the complexity of the Onlooker and Employed bees phase from $\mathcal{O}(n)$ to $\mathcal{O}(n \log n)$ if non-visited components are kept in a separate list for each solution in X in an efficient way

- Changing (2.3) to choose multiple components from $\boldsymbol{x}^i$ would not improve **issue $\#\mathbf{1}$**. Let $J \subset \{1, \ldots, n\}$ and suppose that x^i_j is chosen for each $j \in J$ to update. Update rule (2.3) is an affine transformation in the j-th axis along the line segment between x^i_j and x^k_j, $i \neq k$. If $|J| > 1$, then $|J|$ simultaneous affine transformations in the $|J|$ dimensional subspace between $\boldsymbol{x}^i$ and $\boldsymbol{x}^k$ would be performed. In terms of complexity, there would be no burden if j decision variables are updated at once by means of a matrix product operation. However, in terms of performance, there is no improvement because of two reasons. Firstly, moving along many axes at once does not reduce the possibility of $\boldsymbol{x}^i$ to remain in $\mathcal{N}(\boldsymbol{x}^*)$ if $\boldsymbol{x}^k \in \mathcal{N}(\boldsymbol{x}^*)$. Secondly, setting $|J| > 1$ in (2.3) has been shown to be not as good as $|J| = 1$ in later iterations due to the coarseness of the search when most of the solutions have converged to a single accumulation point [6].

In the following section, we present a method that provides a solution to the issues stated above, the Adaptive Decision Variable Matrix (A-DVM). The A-DVM can be integrated to any optimization technique that uses a set of solution. However, we limit the A-DVM to the Artificial Bee Colony so that we can limit the scope to one algorithm in particular.

3.2 A novel Decision variable mechanism

We propose a method for selecting decision variables efficiently without any additional memory nor simultaneous update of multiple components. The Adaptive Decision Variable Matrix (A-DVM) is an extension of the decision variable selection procedure of Mollinetti et al. [47]. It exploits the same modular nature as the Artificial Bee Colony (ABC), and thereby it can be integrated to the employed and/or onlooker bees phase without interfering with any additional steps of the original or any variant. To emphasize the difference between the A-DVM and Mollinetti et al. [47] deterministic selection, we briefly explain their proposition as follows.

3.2.1 Fully deterministic decision variable selection

The selection scheme proposed by Mollinetti et al. [47] is inspired by Cantor's Diagonalization argument used to prove the non-existence of bijection from the set of natural numbers to the set of real numbers [50, 51]. Cantor's argument state that, any binary square matrix $\boldsymbol{T}$ does not have the same column as the vector consisting of the complements of the diagonal elements of $\boldsymbol{T}$. The authors extended this notion to generate new solutions $\boldsymbol{x}^i$ in the solution set $\boldsymbol{X}$. For any given problem, the deterministic decision variable selection arranges the solution set $\boldsymbol{X}$ into a $\mathbb{R}^{n \times SN}$ matrix:

$$\boldsymbol{A} = [\boldsymbol{x}_1 \; \boldsymbol{x}_2 \; \cdots \; \boldsymbol{x}_{SN}].$$

If $\boldsymbol{A}$ is a square matrix, the entries on the main diagonal are stored in an m-vector $\boldsymbol{c} = (a_{11}, a_{22}, \ldots, a_{mn})^\top$ and undergo the update step. In general, the higher the number of solutions, the better the exploration of the search, and so $SN > n$ holds. If $\boldsymbol{A}$ is wide, then vector $\boldsymbol{c}$ consists of entries on the main diagonal and the superdiagonals of $\boldsymbol{A}$ offset n units to the right. For instance, if $\boldsymbol{A}$ is a 2×6 matrix, then $\boldsymbol{c}$ will be:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \end{bmatrix} \quad \boldsymbol{c} = (a_{11}, a_{22}, a_{13}, a_{24}, a_{15}, a_{26})^\top.$$

The vector $\boldsymbol{c}$ allows (2.3) to be performed simultaneously for all columns of $\boldsymbol{A}$ by means of a simple vector multiplication:

$$\boldsymbol{c}' = \psi(\boldsymbol{c} + \varPhi \odot (\boldsymbol{c} - \boldsymbol{z})), \tag{3.2}$$

where $\odot$ is the Hadamard product, $\varPhi$ is a SN column vector of values sampled from $U(-1,1)$, $\boldsymbol{z} = (z_1, \ldots, z_{SN})^\top \neq \boldsymbol{c}$, and $\psi(\cdot)$ is a function similar to (2.4) defined as:

$$\psi(c_i') = \begin{cases} c_i',\ l_{c_i} = 0 & \text{if } f(\boldsymbol{x}^i + (c_i' - x_j^i)\boldsymbol{e}_j) \leq f(\boldsymbol{x}^i) \\ c_i,\ l_{c_i} = l_{c_i+1} & \text{otherwise.} \end{cases} \tag{3.3}$$

Suppose in the matrix $\boldsymbol{A}$ of the above example, that if $f(\boldsymbol{x}^1 + (c_1' - x_1^1)\boldsymbol{e}_1) \leq f(\boldsymbol{x}^1)$ and $f(\boldsymbol{x}^2 + (c_2' - x_2^2)\boldsymbol{e}_2) \leq f(\boldsymbol{x}^2)$, then $c' = (c_1', c_2', a_{13}, a_{24}, a_{15}, a_{26})^\top$. Thus, entries a_{11} and a_{22} are replaced with c_1', c_2' in $\boldsymbol{A}$, and the corresponding values of f are updated.

Lastly, a safeguard step is performed so that every decision variable of each candidate solution can be updated at least once before the algorithm termination. The last column $\boldsymbol{x}^{SN}$ of $\boldsymbol{A}$ is moved to the first position and the remaining columns are shifted one position to the right. Referring back to the example, the matrix $\boldsymbol{A}$ is now:

$$\boldsymbol{A} = \begin{bmatrix} c_1' & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & c_2' & a_{23} & a_{24} & a_{25} & a_{26} \end{bmatrix} \longrightarrow \begin{bmatrix} a_{16} & c_1' & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{26} & a_{21} & c_2' & a_{23} & a_{24} & a_{25} \end{bmatrix}$$

This step ensures that every decision variable is updated by (3.2) every d iterations.

The results in Mollinetti et al. [47] indicate that eliminating the randomness in the choice of the decision variable in (2.3) boosted the performance of the original ABC in multimodal problems of up to 30 decision variables. However, it is observed that the diversity of solutions was compromised because local search was more emphasized over global search. From this result, we suppose that the bias towards local search brought by the fully deterministic parameter selection has yet not solved **issue #1**. In fact, if anything, the fully deterministic selection made it worse. Therefore, reintroducing a small degree of randomness while guaranteeing that every solution is chosen at some iteration is a step in the right direction to refocus global search.

3.2.2 A self-adaptive decision variable selection procedure (A-DVM)

Let us change the focus to a partially deterministic selection, and reintroduce an adaptive degree of randomness to the selection process based on the ”spread” of solutions throughout the search space. The variables x_j^i are chosen via a binary decision matrix. The goal of the A-DVM is not only to provide an acceptable solution to the issues discussed in section 3.1, but to improve the overall performance of the state-of-the-art of ABC for the multimodal and high dimensional problem of the form of (2.1).

The main piece of the A-DVM is the $n \times SN$ binary matrix $\boldsymbol{P_{am}}$, that represents which x_j^i has been chosen to be updated by (2.3) or (3.2). The matrix $\boldsymbol{P_{am}}$ is a composition of two matrices, $\boldsymbol{P_r}$, a binary matrix with a single 1 in each column, whose row is determined randomly according to a uniform distribution; and $\boldsymbol{P_d}$, a matrix with 0 or 2 in each entry generated by the fully deterministic scheme of Mollinetti et al. [47]. For example, $\boldsymbol{P_r}$ and $\boldsymbol{P_d}$ are matrices of the form:

$$\boldsymbol{P_r} = \begin{pmatrix} 0 & 0 & 0 & \dots & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & 1 \\ 0 & 1 & 0 & \dots & 0 & 0 & \dots & 0 \end{pmatrix}, \qquad \boldsymbol{P_d} = \begin{pmatrix} 2 & 0 & 0 & \dots & 0 & 2 & \dots & 0 \\ 0 & 2 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 0 & 2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 2 & 0 & \dots & 2 \end{pmatrix}.$$

The matrix $\boldsymbol{P_{am}}$ is the result of a composition of $\boldsymbol{P_r}$ into $\boldsymbol{P_d}$. That means some solutions $\boldsymbol{x}_i \in X$ have their j-th component randomly selected when updated by (2.3) or (3.2) while the rest have their j-th component chosen by the fully deterministic scheme. We write $\boldsymbol{P_{am}} = \beta \boldsymbol{P_r} \oplus \alpha \boldsymbol{P_d}$ when $\beta\%$ of the columns of $\boldsymbol{P_{am}}$ are from $\boldsymbol{P_r}$ and the remaining $\alpha = (1-\beta)\%$ are from $\boldsymbol{P_d}$. An example of $\boldsymbol{P_{am}}$ based on the above example is as follows:

$$\boldsymbol{P_{am}} = \beta \boldsymbol{P_r} \oplus \alpha \boldsymbol{P_d} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 2 & 0 & \dots & 0 & 0 & \dots & 0 \\ 1 & 0 & 2 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \dots & 2 & 0 & \dots & 2 \end{pmatrix}.$$

The degree of how much $\boldsymbol{P_r}$ is favored over $\boldsymbol{P_d}$ is represented by the coefficient α to maintain a healthy diversity of solutions while balancing between local search and local search. α is iteratively adjusted as follows,

$$\alpha = (1 - \Delta)K_1 + \Delta K_2, \tag{3.4}$$

where $\Delta \in [0, 1]$ is the measure of the dispersion of the population at the current iteration and K_1 and K_2 are scaling parameters set to 0.3 and 0.7 in accordance to McGinley et al. [52]. Values of α close to 1 signify high population diversity and activate local search by the deterministic selection. On the other hand, values close to 0 boost exploration using random selection. Because solutions in population-based algorithms tend to concentrate around accumulation points $\boldsymbol{x}' \in X'_{f,S}$ after a considerable amount of iterations [49], α is increased by a growth function ρ defined as follows, to intensify local search around $\boldsymbol{x}'$ after t' iterations:

$$\rho(\cdot) = \begin{cases} \alpha = \alpha e^{\gamma t} & \text{if } t > t', \\ \alpha & \text{otherwise}, \end{cases} \tag{3.5}$$

where γ is set to 0.01. The value of t' is given by:

$$t' = \min\left(\frac{n \cdot d}{\lambda_t \cdot t_{max}},\ \lambda_t t_{max}\right), \tag{3.6}$$

where an acceptable value for λ_t was empirically verified to be 0.1.

To ensure that every decision variable is chosen in at least every n iterations, we introduce a history $\boldsymbol{H} \in \{0, 1\}^n$ that stores which columns of $\boldsymbol{P_d}$ were put into $\boldsymbol{P_{am}}$. This is done to allow the remaining columns of $\boldsymbol{P_d}$ that were not chosen in the previous iteration to be included $\boldsymbol{P_{am}}$ at

Algorithm 3: Steps of the A-DVM

```
1  $\mathit{\Delta} \leftarrow$ ComputeDelta($\boldsymbol{X}$)
2  $\alpha \leftarrow \rho\,((1-\mathit{\Delta})K_1 + \mathit{\Delta}K_2)$
3  $\beta \leftarrow 1-\alpha$
4  $\boldsymbol{P_r} \leftarrow$ BuildRandomMatrix($\beta$)
5  $\boldsymbol{P_d} \leftarrow$ BuildDeterministicMatrix($\alpha, \boldsymbol{H}$)
6  if $\boldsymbol{H} = (1,\dots,1)$ then
7      $\boldsymbol{H} \leftarrow (0,\dots,0)$
8  $\boldsymbol{H} \leftarrow$ UpdateHistory($\boldsymbol{H}, \boldsymbol{P_d}, \boldsymbol{P_{am}}$)      ▷ Index of columns of $\boldsymbol{P_d}$ in $\boldsymbol{P_{am}}$ are $1$ in $\boldsymbol{H}$
9  $\boldsymbol{P_{am}} \leftarrow \beta\boldsymbol{P_r} \oplus \alpha\boldsymbol{P_d}$
10 $\boldsymbol{c} \leftarrow$ ChooseVariables($\boldsymbol{X}, \boldsymbol{P_{am}}$)
11 $\boldsymbol{c}' \leftarrow$ UpdateStep($\boldsymbol{c}$)      ▷ Use either (2.3) or (3.2)
12 $\boldsymbol{X} \leftarrow$ UpdateSolutions($\boldsymbol{X}, \boldsymbol{c}'$)
13 $\boldsymbol{X} \leftarrow$ SafeguardStep($\boldsymbol{X}$)
```

the next iteration. We enforce a bound on the number of iterations that solutions are chosen by the fully deterministic selection to be no more than $\frac{3}{5}K_1$ and no less than $\frac{1}{2}K_2$ (refer to (3.4)). When the entries in $\boldsymbol{H}$ are all ones, $\boldsymbol{H}$ is reinitialized and the whole process runs again. The overall steps of the A-DVM are outlined in Algorithm 3.

3.2.3 The $\mathit{\Delta}$ Dispersion Estimate

Estimating the dispersion of the solutions in the search space is specifically effective for population-based algorithms to deal with multimodal or high dimensional problems. Measuring how far apart solutions in $\boldsymbol{X}$ are from each other is very helpful to guide them towards accumulation points or free them from local optima. Significant contributions related to this subject can be found in Ursem [53] and Back et al. [54] which introduced the Sparse Population Diversity (SPD) metric, a method for estimating the variation of the solution set by measuring the distance from each solution in relation to the centroid. McGinley et al. [52] proposed the Healthy Population Diversity (HPD), an extension of the SPD that introduces the concept of individual contribution to the computation of the centroid.

Metrics like SPD and HPD may accurately and inexpensively identify differences between the solutions in $\boldsymbol{X}$ by measuring the distances to each $\boldsymbol{x}^i$. However, this kind of measurement does not take into account how the solutions are distributed in the search space, which is problematic since the same measurement values from SPD and HPD may indicate different search-space coverage of solution of $\boldsymbol{X}$. Because of that, we employ the $\mathit{\Delta}$ dispersion measure introduced in Morrisson [48], initially proposed for Evolutionary Algorithms with binary solution encoding, and adapt it for the continuous problem (2.1). Computation of $\mathit{\Delta}$ is as follows:

$$\mathit{\Delta} = \mathit{\Delta}_1 + \mathit{\Delta}_2 = \frac{1.75 - \mathbb{S}}{1.75}, \tag{3.7}$$

where $\mathit{\Delta}_1 = 0.75 - \mathbb{S}_1$, $\mathit{\Delta}_2 = 1 - \mathbb{S}_2$ and $\mathbb{S} = \mathbb{S}_1 + \mathbb{S}_2$. The values of $\mathbb{S}_1$ and $\mathbb{S}_2$ are obtained by measuring the moment of inertia of the solution centroid in relation to each solution. We denote as $P = |\boldsymbol{X}|$ as the number of solutions $\boldsymbol{x}^i \in \boldsymbol{X}$. The centroid cr_j of the jth components and the moment of inertia I_j of centroid cr_j are

$$cr_j = \frac{\sum_{i=1}^{P} x_j^i}{P}, \qquad I_j = \sum_{i=1}^{P} (x_j^i - cr_j)^2, \quad j = 1, \ldots, n. \tag{3.8}$$

The first measure $\mathbb{S}_1$ involves a quantitative assessment of the solutions around the distribution centroid. Assuming the distribution around the centroid to be uniform, $\mathbb{S}_1$ is

$$\mathbb{S}_1 = \max_j \frac{\left[\left|I_{U_o} - I_j + P cr_j^2\right|\right]}{P}, \tag{3.9}$$

where I_{U_o} represents the inertia of an uniform distribution:

$$I_{U_o} = \sum_{i=1}^{P} \left(\frac{i}{P+1}\right)^2. \tag{3.10}$$

Measure $\mathit{\Delta}_2$ indicates how much the calculation of $\mathit{\Delta}_1$ is misleading when the distribution is not uniform in the search-space, since $\mathit{\Delta}_1$ only verifies non-uniformity along the principal diagonal of the search space. The second measure $\mathbb{S}_2$ is

$$\mathbb{S}_2 = \max\left[\frac{\left|\sum_P \chi_{c+} - \left\lceil\left(\frac{\prod_j (1-cr_j)}{\prod_j \phi_j}\right) P\right\rceil\right|}{P}, \frac{\left|\sum_P \chi_{c-} - \left\lceil\left(\frac{\prod_j (cr_j)}{\prod_j \phi_j}\right) P\right\rceil\right|}{P}\right]. \tag{3.11}$$

where $c- = \{x_j^i \in \boldsymbol{X} | x_j^i \leq cr_j\}, c+ = \{x_j^i \in \boldsymbol{X} | x_j^i \geq cr_j\}$, χ is the characteristic function that returns either 0 or 1 whether a solution belongs to $c-$ or $c+$, respectively, and ϕ_j is in the range between $[l_j, u_j]$, so that $\prod_{j=1}^{n} \phi_j = 1$ for an N-dimensional unit volume.

3.2.4 Remarks on complexity

As for the complexity, SN function evaluations are done in each employed and onlooker bees phase, so the addition of A-DVM preserves the same $2n + 1$ function evaluations per iteration as the classical ABC. The effort to compute the sum of moments of inertia and $\mathit{\Delta}$ dispersion is proportional to the solution set size SN [48]. Updates of solutions (2.3) during the employed or onlooker bees is done one by one in a loop require $\mathcal{O}(n)$ time for the size SN of solution set $\boldsymbol{X}$. On the other hand, offline update (3.2) can be done in a linear time due to the vector multiplication. We recommend (3.2) for parallel versions of the ABC, when MCN is large or the evaluation of $f(\boldsymbol{x})$ is expensive.

Regarding lookup table $\boldsymbol{H}$, it is verified in $\mathcal{O}(n)$ time which columns of $\boldsymbol{P_d}$ were not chosen to be a part of $\boldsymbol{P_{am}}$. Lastly, about binary matrix $\boldsymbol{P_d}$, because the deterministic parameter selection extracts the diagonal of solution set $\boldsymbol{X}$, it is recommended to set $SN \geq n$ to ensure that every decision variable of each solution is chosen in at most n iterations.

Table 3.1: Definition of Benchmark functions.

Name	Dim	Range	Opt.	Name	Dim	Range	Opt.
Bukin06	10	[-10, 0]	0.0	Rosenbrock	30	[-30, 30]	0.0
Cola	17	[-4, 4]	11.7464	Schwefel06	30	[-500,500]	0.0
CrossLegTable	2	[-10,10]	-1.0	SineEnvelope	20	[-500, 500]	0.0
CrownedCross	2	[-10,10]	0.0001	Trefethen	2	[-10,10]	-3.0
Damavandi	2	[0,14]	0.0	Whitley	2	[-10.24, 10.24]	0.0
DeVilliersGlasser02	5	[1, 60]	0.0	XinSheYang03	20	[-500, 500]	0.0
Griewank	30	[-100, 100]	0.0	Zimmerman	2	[0, 100]	0.0
Rastrigin	30	[-5.12, 5.12]	0.0	–	–	–	–

Table 3.2: Properties of the benchmark functions.

Name	**Property**	**Name**	**Property**
Bukin06	CNdNsNscMm	Rosenbrock	CDNsScMm
Cola	CDNsNscMm	Schwefel06	CDSNscMm
CrossLegTable	CNsNscMm	SineEnvelope	CDNsScMm
CrownedCross	CNsNscMm	Trefethen	CDNsNscMm
Damavandi	CDNsNscMm	Whitley	CDNsScMm
DeVilliersGlasser02	CDNsNscMm	XinSheYang03	CNdNsNscMm
Griewank	CDNsScMm	Zimmerman	CNdNsScMm
Rastrigin	CNdNsScMm	-	

3.3 Numerical Experiments

We carry several numerical experiments to answer the following research questions: ”Does the incorporation of the Adaptive Decision Variable Matrix (A-DVM) to any version of the Artificial Bee Colony (ABC) incur in improvements in the overall performance to solve multimodal optimization problems?”; ”Does the A-DVM scales well on instances where $n > 100$?”; and ”Can an A-DVM based ABC variant obtain competitive results in the same instances when compared to other derivative-free methods?”. We seek to answer these three questions with three distinct experiments using 15 unconstrained functions in the form of (2.1), each of which is designed to validate the capability of derivative-free algorithms to handle multimodal and non-smooth objective functions. The instances are ranked in the top 30 hardest continuous optimization functions in the Global Optimization Benchmarks suite [55]. The number of variables, the box constraint range $[l_j, u_j]$ and the global optimum of each instance are listed in Table 3.1.

The classification of the properties of each function is shown below according to the classification of Jamil [56], Molga and Smutinicki [57] and [58]. We classify each function as: Differentiable (**D**) or non-differentiable (**Nd**); continuous (**C**) or non-continuous (**Nc**); Separable (**S**) or Non-separable (**Ns**); Scalable (**Sc**) or Non-scalable (**Nsc**); Multimodal (**Mm**) or Unimodal (**Um**).

The experiments were conducted in a machine with the following hardware configuration: Intel core i7-6700 ”Skylake” 3.4 GHz CPU; 16 GB RAM DDR4 3200 clocked at 3000 MHz. The running operating system (OS) is UbuntuOS 18.04. All algorithms were written in the python 3 programming language. Floating point operations were handled by the numpy package, version 1.19.1.

3.3.1 Testing the A-DVM in the Artificial Bee Colony algorithm

The answer for the first question is backed up by an experiment incorporating the A-DVM to the onlooker and employed bees phase of the following versions of the ABC: the original ABC from Karaboga [59] (ABC+A-DVM), two versions of the global best guided ABC (gbestABC) from Gao et al. [60] (GBESTABC+A-DVM, GBESTABC2+A-DVM) and two versions of the ABC-X from [6] for multimodal problems (ABC-XM1+A-DVM, ABC-XM2+A-DVM). The original counterparts were also used for the baseline (ABC, GBESTABC, GBESTABC2, ABC-XM1, ABC-XM5) together with the modified ABC for multidimensional functions (MABC) from Akay and Karaboga [8] and its version with the A-DVM (MABC+A-DVM). We also included three well-known population heuristics as baselines of comparison, the Particle Swarm Optimization from Kennedy and Eberhart [61], Evolutionary Particle Swarm Optimization by Miranda and Fonseca [62] and Differential Evolution (DE) [63].

The Stopping criteria for each algorithm was set to 10^5 function evaluations (FE's) or if the difference between the best value found so far and the global optimum $f(\boldsymbol{x}^*)$ is less than 10^{-8}. The population size was common to all algorithms and fixed at 30. For PSO, the inertia factor (w_1) was set to 0.6 and both cognitive and social parameters (w_2, w_3) to 1.8. For Differential Evolution (DE) [63] with *best1bin* strategy, *F* value was 0.5 and *CR* 0.9. For each version of the ABC: $Lit = SN \cdot n$. For MABC, MR, SF and m were 0.5, 0.7, and 2.5% of maximum FE's, respectively. ABC-X parameters were $Lit = 1.06 \cdot n$, maximum population of 66 and minimum of 15 for ABC-Xm1 and $Lit = 0.83 \cdot n$, while for ABC-Xm5, maximum population of 78 and minimum of 17. Lastly, parameters γ and λ_t of the A-DVM were set to 0.1. Each algorithm is executed 30 times with the same seed interchangeably in a random fashion to avoid bias in the machine load. Figure 3.1 show the percentage of the best mean of the runs for each algorithm and the rank of each algorithm for each family. In this categorization, the worst rank that an algorithm can have is the 6th, while the best is the 1st.

Tables 3.7 and 3.8 show the computational results obtained from this experiment. The statistics used for comparison are the mean, standard deviation, median, and best-worst results obtained from 30 runs with distinct random seeds. Statistical significance between pairs is verified by the Mann-Whitney U test for non-parametric data, with confidence interval α set to 0.95. For better legibility, the precision of decimals is set to 5 digits and values lower than 10^{-6} are rounded to 0. Plots of the behavior of each algorithm are shown in Figure 3.2, 3.3 and 3.4. Each line represents the mean of the best solution of all executions for each function evaluation call. All plots were log-scaled for better legibility.

Statistical analysis is performed using the Mann-Whitney U-test for non-parametric data. For the sake of brevity, we supplement the p-value tables for each instance in the Appendix section (Table 3.9 to 3.12). If the performance of any algorithm for a particular instance is statistically significant, it means that its p-value in the U test is less than 0.05 in the pairwise comparison against all other algorithms. The bold numbers in the tables indicate the least value for that particular statistic and instance.

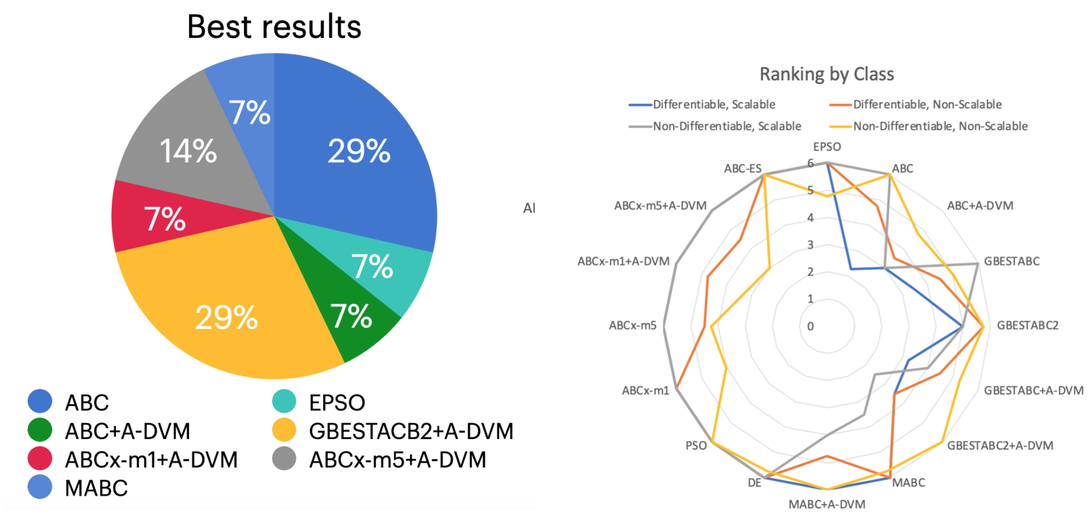


Figure 3.1: Percentage of the best overall mean and radar plot of the ranking of each technique according to family.

Firstly, we discuss the Rosenbrock, Whitley and Zimmerman function instances where the A-DVM resulted in overall worse performance than all of their original counterparts. The A-DVM was indeed able to guide the functions towards a valley, but a thorough local search mechanism was lacking due to the parabolic surface of the Rosenbrock function [57]. The same behavior is observed in the Whitley and Zimmerman functions, which share the same property as Rosenbrock instance. The poor results in these functions imply a failure of the A-DVM to properly address issues **#1** and **#2** discussed in section 3.1. Additionally, we can relate this case to the no-free-lunch theorem of Wolpert [64], saying that no algorithm can be strictly better than the others in every problem instance. The inferior results of the A-DVM are also seen for the Rastrigin function in the ABC-X variants. The cause of such behavior could be due to intensification of the local search mechanism that forced solutions to stay far from the local attractors of the surface of the functions. Overall, it was observed that the A-DVM based algorithms did not perform well in Non-differentiable instances.

On the other hand, it was possible to observe a strong evidence of the robustness of the A-DVM in Differentiable, Non-separable and such as the Damavandi, DeVilliersGlasser02 and CrossLegTable instances, ranked as the three hardest functions in the benchmark suite [55]. Both functions feature large basins of attraction for bad local optima, the number of which is directly proportional to the problem dimensionality. There are two possible causes explaining why the A-DVM versions were not superior to all other versions in these particular instances. First, a small number of dimensions means that a square matrix can be built, providing a thorough exploration of the search space. Second, exploration in the early stages allowed solutions to escape from the basins of attraction.

Evidence that the A-DVM improved the search process in comparison to their counterparts without the search can be seen in the Bukin06, SineEnvelope, CrownedCross and Schwefel06. Although the ABCs with the A-DVM were not the best solvers, their robustness was statistically

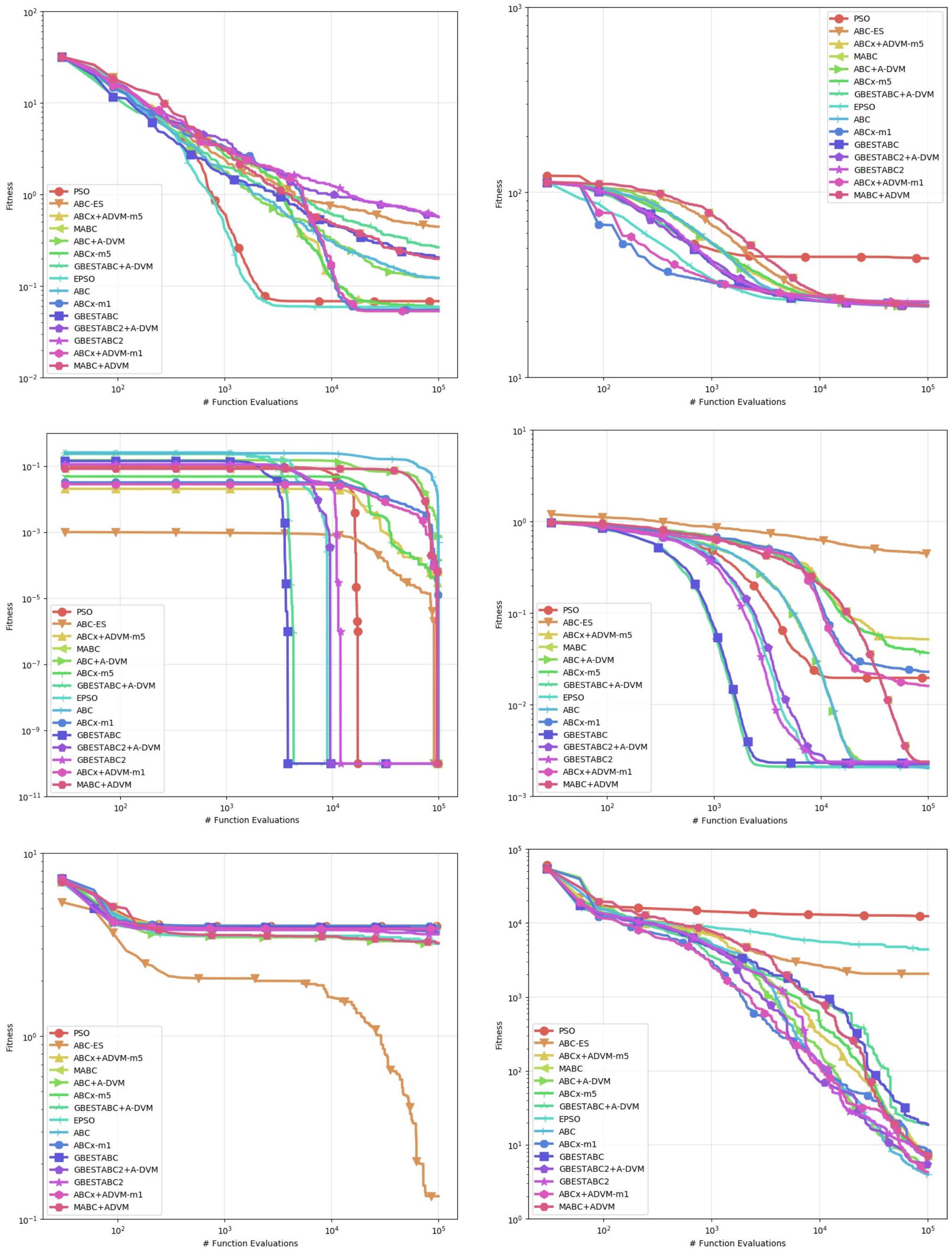


Figure 3.2: Behavior of the Mean of all executions for Bukin06 to DeVilliersGlasser02 instances.

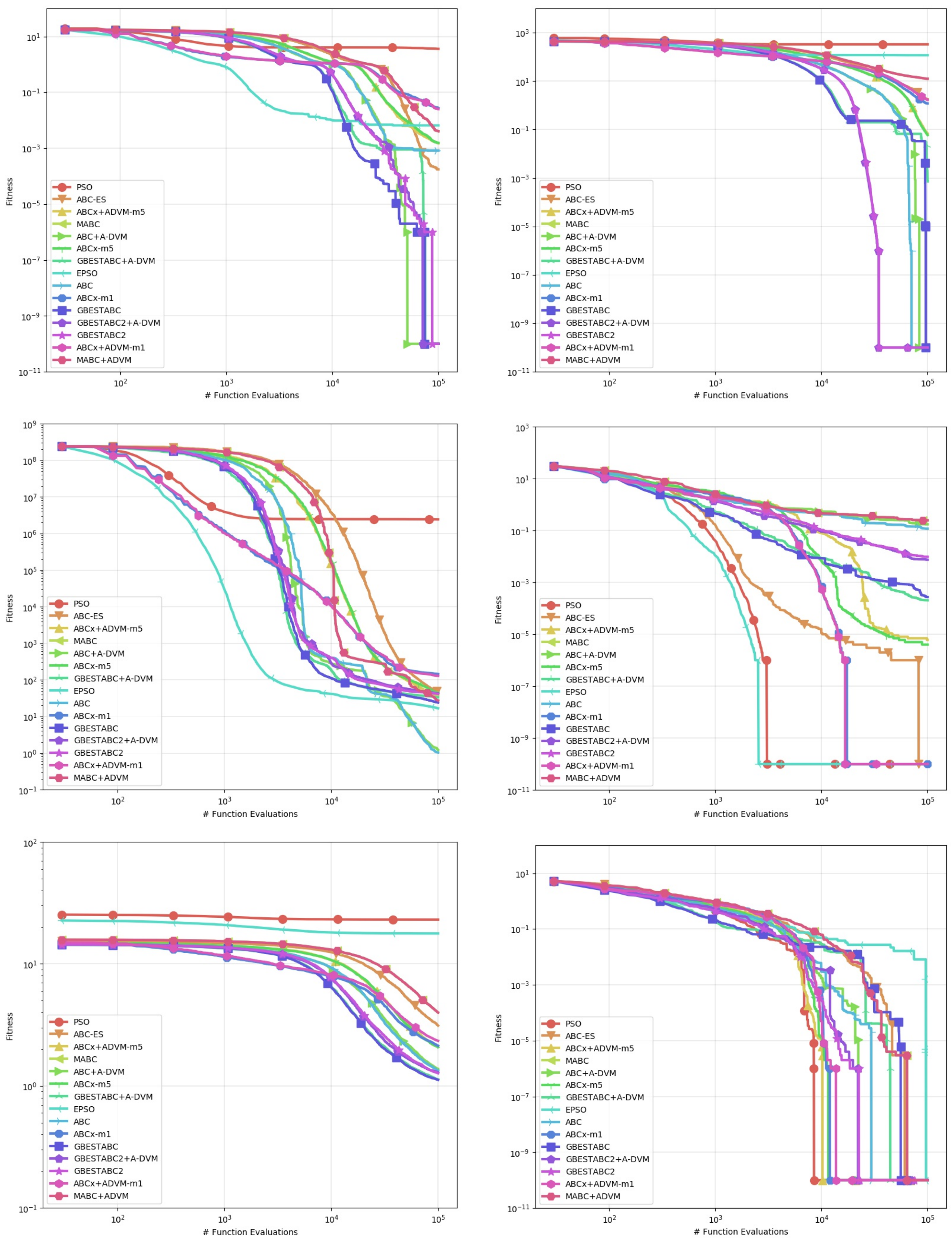


Figure 3.3: Behavior of the Mean of all executions for Griewank to Trefethen instances.

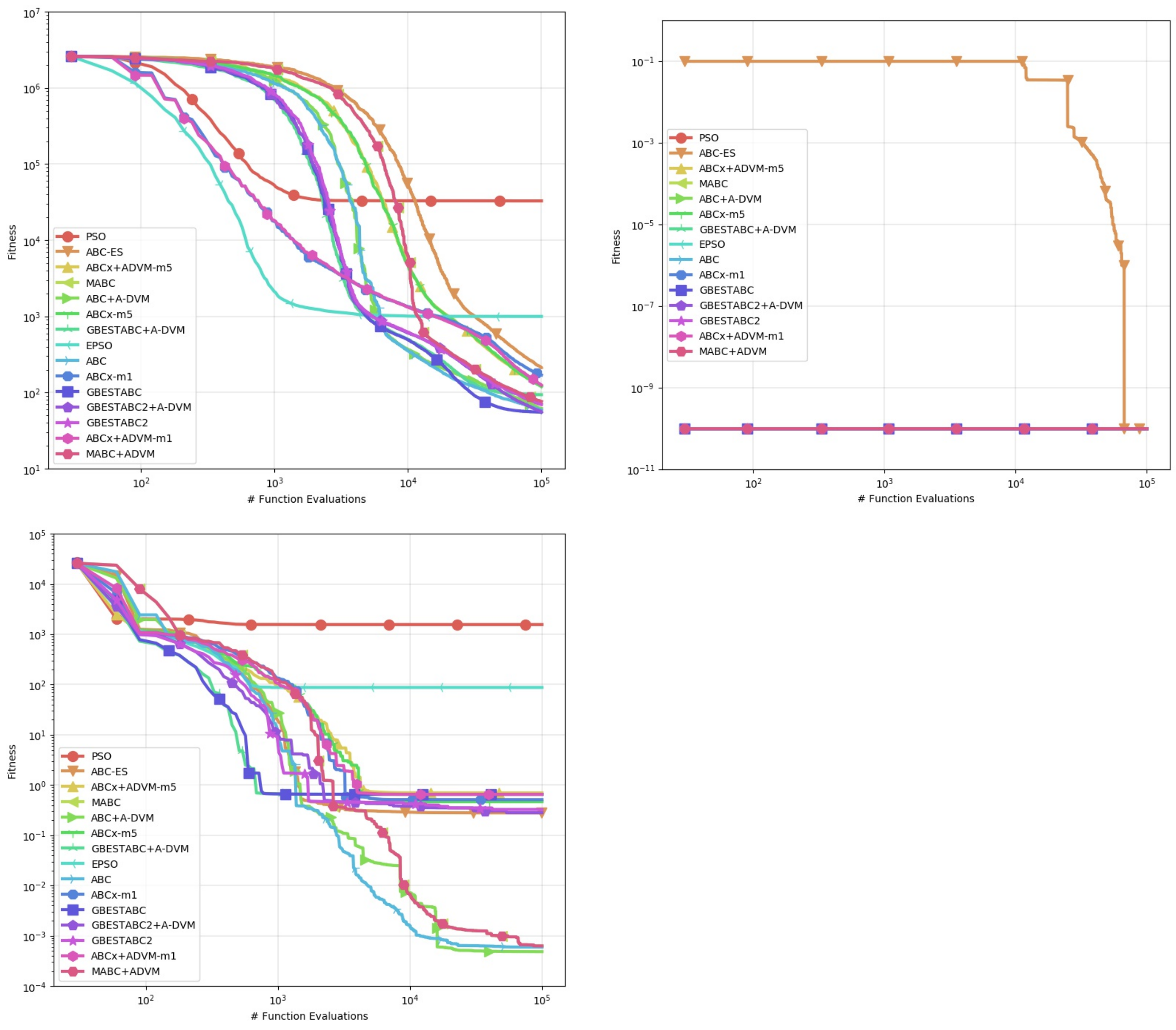


Figure 3.4: Behavior of the Mean of all executions for Whitley to Zimmerman instances.

significant in comparison to the versions without the A-DVM. Lastly, in the Cola, Griewank, XinSheYang03 and Trefethen, no statistical significance that corroborated that the incorporation of the A-DVM improved or worsened the performance of the original algorithm was found, once again pointing to the weakness of the A-DVM in non differentiable family of problems.

3.3.2 Testing the A-DVM in large scale instances

In order to answer whether the A-DVM is able to improve the robustness of the base ABC algorithms in large multimodal instances, another numerical experiment is carried out. We chose a subset comprised of the scalable functions of Table 3.1, and set their dimension to 100, as shown in Table 3.3.

Table 3.3: Subset of scalable functions.

Name	Dim	Range	Opt	Property
Griewank	100	[-10, 0]	0.0	CDNsScMm
Rastrigin	100	[-4, 4]	11.7464	CNdNsScMm
Rosenbrock	100	[-10,10]	-1.0	CDNsScMm
Sine Envelope	100	[-10,10]	0.0001	CDNsScMm
Whitley	100	[0,14]	0.0	CDNsScMm
Zimmerman	100	[1, 60]	0.0	CNdNsScMm

As in section 3.3.1, we use the the original ABC, GBESTABC, GBESTABC2 and two versions of the ABC-X (ABCX-m1, ABCX-m5). The original counterparts were also used for the baseline (ABC, GBESTABC, GBESTABC2, ABCX-m1, ABCX-m5) together with the modified ABC for multidimensional functions (MABC) from Akay and Karaboga [8] and its version with the A-DVM (MABC+A-DVM). In this experiment, we limit the comparison to ABC-based algorithms only for a fairer and more in-depth analysis. The parameters defined in this experiment is the same as in Section 3.3.1 with the difference that the Stopping criteria for each algorithm was set to 10^4 function evaluations (FE's). Figure 3.5 show the percentage of the best mean of the runs for each algorithm and the rank of each algorithm for each family in the same fashion as the previous section.

The same statistics from the last section (Mean, Median, Standard Deviation, Best and Worst execution) are chosen to show the results. Overall statistical differences are ascertained by the Friedman test while pairwise differences are verified by the Mann-Whitney U test, both with their confidence α set to 95%. The statistics of the six instances are shown in Table 3.4 and 3.5, Friedman test and Mann-whitney U-test p-values are displayed in Tables 3.6, 3.13 and 3.14. Log-scaled mean of the 30 executions for each instance throughout the iterations are recorded in Figure 3.6.

Overall statistical difference was observed in every problem since the p-values of table 3.6 were lower than 0.05, therefore each instance will be analyzed individually. We begin stating that with exception of the Zimmerman function, the ABCX-m1 and its A-DVM version were observed to be better suited to large-scaled problem by a large margin, since pairwise differences against all others were seen to be relevant ($p < 0.05$). As for the Zimmerman function, curiously the classical

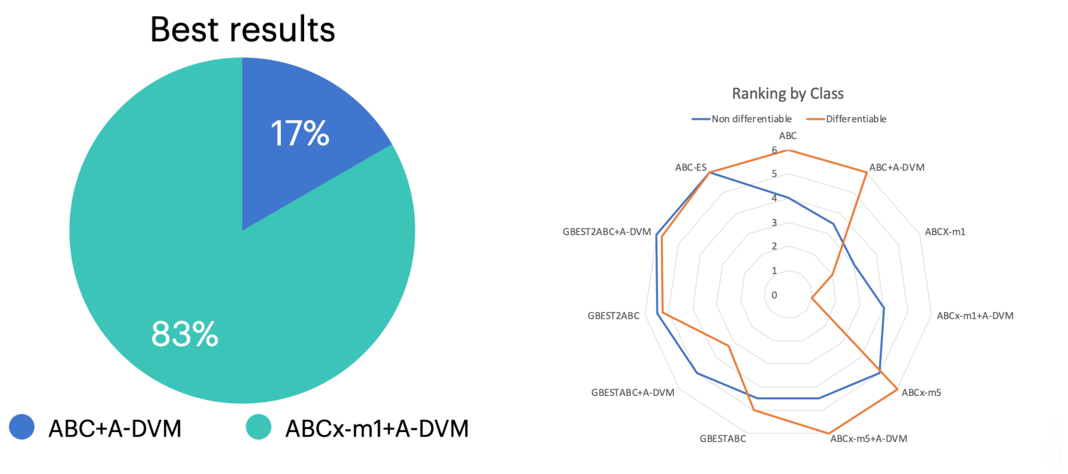


Figure 3.5: Percentage of the best overall mean and radar plot of the ranking of each technique according to family for the second experiment.

Table 3.4: Results of the Griewank, Rastrigin and Rosenbrock instances.

Problem	**Algorithm**	**Mean**	**Median**	**Std. Dev**	**Best**	**Worst**
Griewank	GBESTABC2	22.2549	22.078	1.24253	20.209	24.8705
	GBESTABC	19.1578	19.2894	1.41349	15.8547	22.2885
	GBESTABC+A-DVM	19.165	19.46	1.40259	15.2549	21.2983
	ABC+A-DVM	33.0033	32.5923	3.00956	27.7354	39.3985
	ABC	33.6735	34.2599	3.37994	25.0454	39.5429
	ABCX-m1+A-DVM	**2.35473**	**2.37487**	**0.141961**	**2.09558**	**2.5906**
	ABCX-m5+A-DVM	35.7663	36.0063	1.59183	30.9719	39.6456
	MABC+A-DVM	49.0362	49.5908	3.86792	40.5036	56.2519
	GBESTABC2+A-DVM	22.604	22.5136	1.46622	19.5191	25.5375
	MABC	49.0362	49.5908	3.86792	40.5036	56.2519
	ABCx-m1	2.36628	2.36703	0.202239	2.04665	2.81848
	ABCx-m5	35.7127	35.6153	1.88481	31.8163	39.3963
	ABC-ES	33.1001	28.9053	17.5708	10.4031	63.0538
Rastrigin	GBESTABC2	927.908	927.712	35.9572	860.483	1000.53
	GBESTABC	882.214	879.961	21.3248	848.565	942.571
	GBESTABC+A-DVM	877.636	882.174	28.3145	833.021	923.515
	ABC+A-DVM	992.844	997.666	42.586	848.59	1070.25
	ABC	989.606	984.591	44.6942	863.339	1065.21
	ABCX-m1+A-DVM	370.099	368.148	30.2616	310.793	450.279
	ABCX-m5+A-DVM	1082.16	1085.06	34.4644	1004.84	1137.48
	MABC+A-DVM	1238.44	1243.78	44.9048	1151.41	1308.85
	GBESTABC2+A-DVM	939.86	944.664	31.6091	871.653	990.464
	MABC	1238.44	1243.78	44.9048	1151.41	1308.85
	ABCx-m1	**363.668**	**358.431**	**24.1683**	**326.521**	**419.939**
	ABCx-m5	1098.85	1099.48	25.7644	1044.91	1145.22
	ABC-ES	972.865	923.329	226.278	653.649	1323.36
Rosenbrock	GBESTABC2	1.86248e+08	1.83617e+08	3.62851e+07	1.13022e+08	2.68372e+08
	GBESTABC	1.50072e+08	1.47095e+08	3.02568e+07	8.64117e+07	2.13212e+08
	GBESTABC+A-DVM	1.45601e+08	1.38338e+08	3.83353e+07	6.05437e+07	2.29965e+08
	ABC+A-DVM	3.86399e+08	3.82692e+08	8.56451e+07	1.74091e+08	5.44036e+08
	ABC	3.95541e+08	4.1348e+08	7.97546e+07	1.50902e+08	5.27773e+08
	ABCX-m1+A-DVM	**743141**	**700600**	**164629**	**415046**	**1.08762e+06**
	ABCX-m5+A-DVM	4.58306e+08	4.60116e+08	5.75986e+07	3.36762e+08	5.62055e+08
	MABC+A-DVM	7.1178e+08	7.00777e+08	8.8605e+07	5.13154e+08	8.99771e+08
	GBESTABC2+A-DVM	1.87117e+08	1.98131e+08	3.23815e+07	1.29873e+08	2.38375e+08
	MABC	7.1178e+08	7.00777e+08	8.8605e+07	5.13154e+08	8.99771e+08
	ABCx-m1	840767	842535	201859	488427	1.31484e+06
	ABCx-m5	4.59397e+08	4.58344e+08	4.27048e+07	3.62077e+08	5.52818e+08
	ABC-ES	4.36908e+08	3.17419e+08	3.34933e+08	4.53576e+07	1.06358e+09

Table 3.5: Results of the Sine Envelope, Whitley and Zimmerman instances.

Problem	**Algorithm**	**Mean**	**Median**	**Std. Dev**	**Best**	**Worst**
SineEnvelope	GBESTABC2	44.3438	44.3044	0.376315	43.4559	45.0189
	GBESTABC	43.9496	44.0291	0.543202	42.8189	44.9741
	GBESTABC+A-DVM	43.8262	43.8863	0.49638	42.7754	44.7082
	ABC+A-DVM	44.367	44.4289	0.555319	42.7661	45.3687
	ABC	44.425	44.5275	0.537809	42.9573	45.447
	ABCX-m1+A-DVM	**30.0062**	**29.8985**	**1.15081**	**27.6261**	**32.6375**
	ABCX-m5+A-DVM	45.1242	45.1754	0.421989	44.1659	45.8359
	MABC+A-DVM	45.9901	46.0908	0.44163	44.7613	46.6727
	GBESTABC2+A-DVM	44.2829	44.2682	0.536358	43.2124	45.2068
	MABC	45.9901	46.0908	0.44163	44.7613	46.6727
	ABCx-m1	30.3508	30.3807	0.911687	28.1017	31.8109
	ABCx-m5	45.0936	45.1811	0.415104	44.3154	45.835
	ABC-ES	43.7584	44.3359	2.09874	39.3983	46.4233
Whitley	GBESTABC2	6.84694e+06	7.07747e+06	1.2699e+06	4.10342e+06	9.48787e+06
	GBESTABC	5.61494e+06	5.69385e+06	1.01168e+06	3.44293e+06	8.1876e+06
	GBESTABC+A-DVM	5.50949e+06	5.52107e+06	953260	2.96255e+06	7.06721e+06
	ABC+A-DVM	1.38938e+07	1.39201e+07	2.70429e+06	7.15176e+06	1.94246e+07
	ABC	1.37228e+07	1.40827e+07	2.96759e+06	5.29784e+06	1.98931e+07
	ABCX-m1+A-DVM	**49545.3**	**48046.6**	**5931.77**	**40889.9**	**64243**
	ABCX-m5+A-DVM	1.65758e+07	1.69092e+07	1.39608e+06	1.39002e+07	1.93659e+07
	MABC+A-DVM	2.44575e+07	2.43207e+07	3.23327e+06	1.8017e+07	3.14924e+07
	GBESTABC2+A-DVM	7.01378e+06	7.00248e+06	1.01941e+06	4.61244e+06	8.99792e+06
	MABC	2.44575e+07	2.43207e+07	3.23327e+06	1.8017e+07	3.14924e+07
	ABCx-m1	53358.1	55848.9	6581.91	37192.5	64768.1
	ABCx-m5	1.60252e+07	1.59958e+07	1.58795e+06	1.28247e+07	1.8947e+07
	ABC-ES	1.54652e+07	1.16363e+07	1.19375e+07	1.76703e+06	3.63311e+07
Zimmerman	GBESTABC2	0.395896	0.698584	0.34618	1.4e-07	0.698955
	GBESTABC	0.349291	0.349291	0.349291	0	0.698581
	GBESTABC+A-DVM	0.326005	0	0.348514	0	0.698581
	ABC+A-DVM	**0.0153467**	**0.00966524**	**0.0144522**	**0.00065971**	**0.0558293**
	ABC	0.0154458	0.0132182	0.00876252	0.00162515	0.0411998
	ABCX-m1+A-DVM	0.326094	0.00041285	0.348528	1e-06	0.698947
	ABCX-m5+A-DVM	0.233216	0.00030032	0.329297	8.85e-06	0.6997
	MABC+A-DVM	3.4315	0.094002	12.1792	0.023764	57.0393
	GBESTABC2+A-DVM	0.302766	9.989e-05	0.346171	3.2e-07	0.698866
	MABC	3.4315	0.094002	12.1792	0.023764	57.0393
	ABCx-m1	0.256183	5.474e-05	0.336645	3.22e-06	0.698711
	ABCx-m5	0.235636	0.00021179	0.327997	7.47e-06	0.699713
	ABC-ES	2.35559	0.0157106	8.83164	0.00202546	40.1071

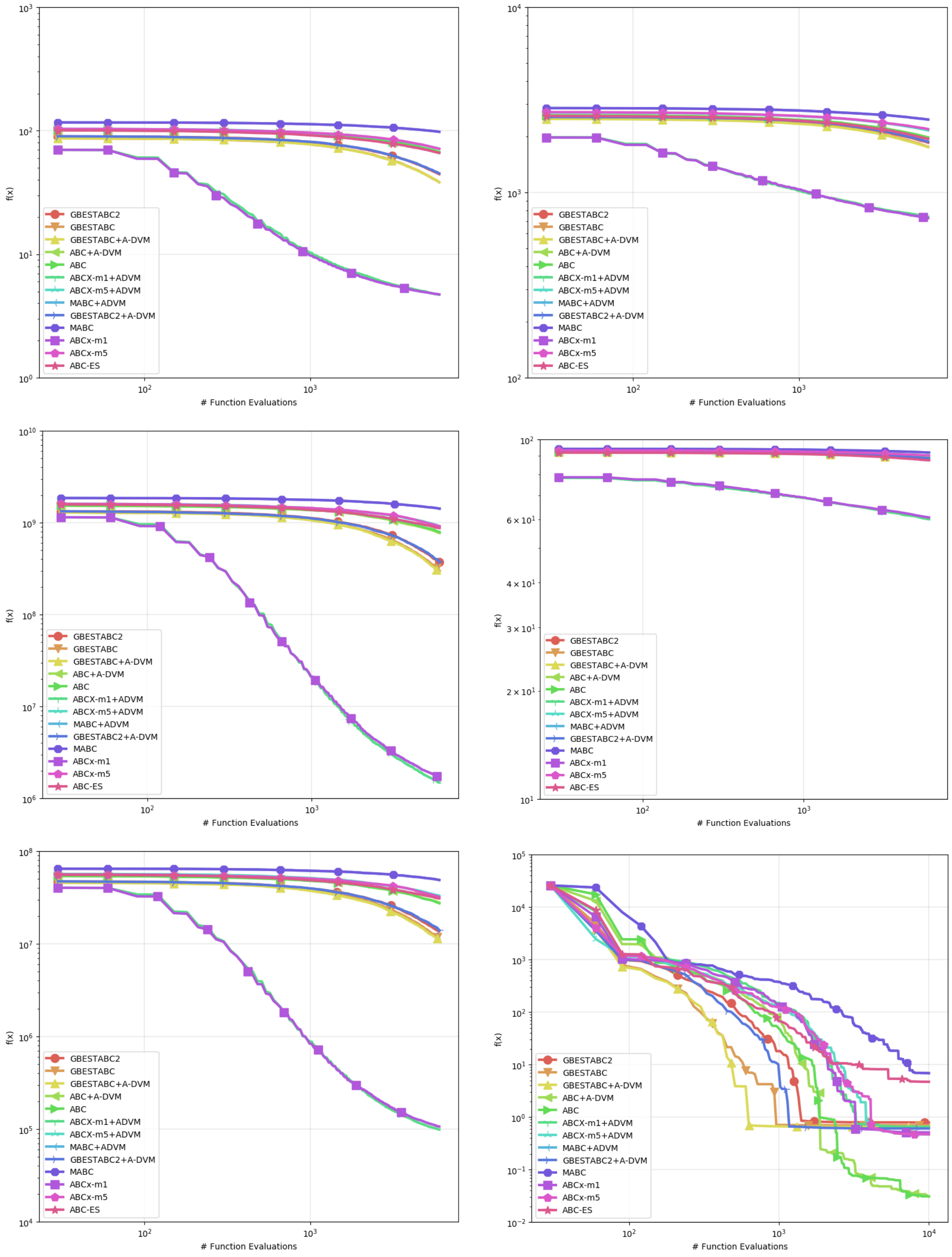


Figure 3.6: Behavior of the Mean of all executions of the scalable instances.

Table 3.6: p-values of the Friedman Test for each instance.

Griewank	Rosenbrock	Rastrigin	SineEnvelope	Whitley	Zimmerman
4.514277e-59	2.078677e-58	7.743842e-59	1.791833e-50	3.497370e-58	0.000003

Table 3.7: Results of the experiment for all problem instances

Problem	**Algorithm**	**Mean**	**Median**	**Std. Dev**	**Best**	**Worst**	**Problem**	**Algorithm**	**Mean**	**Median**	**Std. Dev**	**Best**	**Worst**	**Problem**	**Algorithm**	**Mean**	**Median**	**Std. Dev**	**Best**	**Worst**
Bukin06	DE	0.97893	0.91922	0.52393	0.33	2.60682	DeVilliersGlasser02	DE	352.642	64.37420	767.121	0.39534	3182.44	SineEnvelope	DE	0.47414	0.47354	0.09202	0.3289	0.69363
	PSO	0.03759	0.04883	0.01611	0.00431	0.05000		PSO	7108.73	9377.49	4258.06	**0.00000**	10647.4		PSO	7.09257	7.26654	0.57119	6.09679	7.86571
	EPSO	**0.00901**	**0.00715**	**0.00690**	0.00057	**0.02419**		EPSO	799.059	6.52004	2538.22	**0.00000**	10467.8		EPSO	4.74872	4.8902	0.89428	3.04664	6.09835
	ABC	0.06535	0.05000	0.03345	0.01909	0.15020		ABC	5.71168	3.20875	6.02096	0.34329	21.70150		ABC	**0.25965**	**0.22372**	0.07091	0.18175	0.39453
	MABC	0.05800	0.05000	0.046600	0.01611	0.21930		MABC	4.12467	2.74194	4.63619	0.24565	15.8204		MABC	3.01514	3.09304	0.30318	2.47955	3.57755
	MABC+ADVM	0.099097	0.071473	0.059937	0.013819	0.261345		MABC+ADVM	3.54775	2.48224	3.45717	0.432356	17.9148		MABC+ADVM	1.99107	1.99038	0.332486	1.22778	2.5769
	GBESTABC	0.18781	0.18014	0.07922	0.05044	0.34700		GBESTABC	8.13107	4.71285	11.61610	0.84988	53.06630		GBESTABC	0.25933	0.23034	0.06796	**0.16192**	0.38841
	GBESTABC2	0.35783	0.33525	0.17333	0.12420	0.63954		GBESTABC2	5.56375	4.42650	4.64582	0.65744	21.59900		GBESTABC2	0.30696	0.30139	0.07077	0.19638	0.43203
	ABC+A-DVM	0.05949	0.04990	0.04087	0.00340	0.15985		ABC+A-DVM	2.53581	1.93048	**2.00250**	0.05803	**7.09280**		ABC+A-DVM	0.30694	0.29637	0.09403	0.19931	0.56087
	GBESTABC+A-DVM	0.17330	0.13761	0.08993	0.04872	0.33291		GBESTABC+A-DVM	6.89254	4.77318	5.30292	1.06621	23.2227		GBESTABC+A-DVM	0.26499	0.25378	**0.05339**	0.18960	**0.37096**
	GBESTABC2+A-DVM	0.32394	0.34000	0.17225	0.09892	0.69858		GBESTABC2+A-DVM	8.36485	7.13058	8.01541	0.40725	32.1687		GBESTABC2+A-DVM	0.29561	0.29025	0.05581	0.19367	0.39693
	ABCXm1	0.027548	0.023434	0.017643	0.00241	0.050081		ABCXm1	3.8484	0.338253	11.4397	5.2e-05	59.3269		ABCXm1	1.06673	1.02163	0.344164	0.475796	1.80824
	ABCXm1+A-DVM	0.026575	0.026393	0.018326	**0.000473**	0.050066		ABCXm1+A-DVM	**2.12372**	**0.338253**	3.37561	4.4e-05	12.0782		ABCXm1+A-DVM	1.1684	1.06626	0.366067	0.514902	1.91655
	ABCXm5	0.029684	0.028866	0.014425	0.002358	0.053062		ABCXm5	3.39312	2.46051	2.57071	0.478018	10.445		ABCXm5	1.03955	1.02833	0.16000	0.691504	1.48731
	ABCXm5+A-DVM	0.029074	0.028134	0.016507	0.00326	0.052951		ABCXm5+A-DVM	3.70145	3.39775	2.66626	0.598359	10.0822		ABCXm5+A-DVM	1.04225	1.06835	0.156475	0.760775	1.33566
Cola	DE	12.44390	12.39020	0.502836	11.77570	13.82660	Griewank	DE	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	Trefethen	DE	-3.29379	**-3.30687**	0.04156	**-3.30687**	-3.14408
	PSO	16.13410	15.30270	2.44014	12.9697	22.06270		PSO	1.34282	1.30004	0.17168	1.10178	1.80049		PSO	-3.08315	-3.17611	0.21692	**-3.30687**	-2.64262
	EPSO	13.42210	13.60100	1.06166	**11.7481**	15.48070		EPSO	0.00910	**0.00000**	0.01379	**0.00000**	0.04426		EPSO	-3.27985	**-3.30687**	0.06253	**-3.30687**	-3.06263
	ABC	**12.05440**	**11.95500**	**0.22993**	11.75370	**12.54730**		ABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABC	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	MABC	12.83970	12.84790	0.443416	12.11390	13.61120		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		MABC	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	MABC+ADVM	12.2932	12.3446	0.327254	11.8035	12.9642		MABC+ADVM	0.002013	0.000020	0.00483	**0.00000**	0.022877		MABC+ADVM	-3.30687	-3.30687	**0.00000**	-3.30687	-3.30687
	GBESTABC	12.22790	12.15260	0.30889	11.79990	13.08850		GBESTABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	GBESTABC2	12.23520	12.25250	0.28235	11.80430	12.76960		GBESTABC2	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC2	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	ABC+A-DVM	12.15250	12.08500	0.232509	11.84320	12.65390		ABC+A-DVM	0.00041	**0.00000**	0.00186	**0.00000**	0.00834		ABC+A-DVM	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	GBESTABC+A-DVM	12.28230	12.18300	0.32599	11.81830	13.12510		GBESTABC+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC+A-DVM	**-3.30682**	**-3.30687**	0.00022	**-3.30687**	-3.30588
	GBESTBBC2+A-DVM	12.23630	12.21250	0.29413	11.82890	12.96570		GBESTABC2+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	1.43E-05		GBESTABC2+A-DVM	**-3.30687**	**-3.30687**	**0.00000**	**-3.30687**	**-3.30687**
	ABCXm1	12.7074	12.5568	0.759207	11.7718	14.6546		ABCXm1	0.013916	0.013824	0.01168	4.2e-05	0.047391		ABCXm1	-3.24238	-3.20814	0.070179	-3.30687	-3.06263
	ABCXm1+A-DVM	12.8568	13.0618	0.570437	11.8139	14.1411		ABCXm1+A-DVM	0.012559	0.012324	0.010067	1.5e-05	0.037476		ABCXm1+A-DVM	-3.26056	-3.30687	0.079556	-3.30687	-3.06263
	ABCXm5	12.1119	11.9955	0.262208	11.77	12.7076		ABCXm5	0.000775	0.000684	0.000403	0.000213	0.002204		ABCXm5	-3.26755	-3.30687	0.068596	-3.30687	-3.06263
	ABCXm5+A-DVM	12.0595	12.0312	**0.195371**	11.7971	12.5531		ABCXm5+A-DVM	0.000751	0.000633	0.000424	0.000207	0.002318		ABCXm5+A-DVM	-3.297	-3.30687	0.029619	-3.30687	-3.20814
CrossLegTable	DE	-0.26326	-0.08477	0.37832	**-1.00000**	-0.00611	Rastrigin	DE	0.54722	0.49748	0.60175	**0.00000**	1.98992	XinSheYang03	DE	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	PSO	-0.09723	-0.08283	0.21626	**-1.00000**	-0.00255		PSO	125.84700	126.20500	21.16520	88.1972	160.206		PSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	EPSO	-0.17534	-0.08477	0.28203	**-1.00000**	-0.07959		EPSO	45.76950	51.25510	23.05110	6.96471	99.49550		EPSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABC	-0.13062	-0.08493	0.20463	**-1.00000**	-0.08477		ABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	MABC	-0.10630	-0.08477	0.09595	-0.51398	-0.08477		MABC	74.55290	75.18170	8.78129	50.39630	87.9141		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	MABC+ADVM	-0.084594	-0.084778	0.000587	-0.084933	-0.082837		MABC+ADVM	6.29867	6.04595	1.80949	2.15974	10.3941		MABC+ADVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC	-0.12994	-0.08477	0.20479	**-1.00000**	-0.07981		GBESTABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC2	-0.08448	-0.08477	**0.00071**	-0.08477	-0.08283		GBESTABC2	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC2	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABC+A-DVM	-0.13061	-0.08493	0.20463	**-1.00000**	-0.08477		ABC+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABC+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC+A-DVM	-0.12355	-0.08477	0.20728	**-1.00000**	-0.00656		GBESTABC+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC2+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	GBESTABC2+A-DVM	-0.13044	-0.08477	0.20467	**-1.00000**	-0.08283		GBESTABC2+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		GBESTABC+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABCXm1	-0.032665	-0.039932	0.022017	-0.063706	-0.004235		ABCXm1	0.598405	0.171957	0.709102	0.002089	2.10224		ABCXm1	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABCXm1+A-DVM	-0.028716	-0.041177	0.019874	-0.055539	-0.003221		ABCXm1+A-DVM	0.833917	0.824083	0.842761	0.004411	3.21893		ABCXm1+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABCXm5	-0.049196	**-0.006641**	0.178485	**-1.00000**	-0.001788		ABCXm5	0.029805	0.028714	0.009302	0.014585	0.05643		ABCXm5	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	ABCXm5+A-DVM	**-0.020866**	-0.007002	0.031149	-0.084778	-0.000575		ABCXm5+A-DVM	0.034629	0.029712	0.017951	0.01176	0.091563		ABCXm5+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**

Table 3.8: Results of the experiment for all problem instances

Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst	Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst	Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst
CrownedCross	DE	0.00173	0.00117	0.00347	**0.00010**	0.01635	Rosenbrock	DE	32.86880	25.58250	24.11440	7.49392	86.07950	Whitley	DE	0.01388	1.20E-05	0.01925	**0.00000**	0.03945
	PSO	0.01609	0.00120	0.01884	**0.00010**	0.03909		PSO	163741	122694	107137	31992.8	449777		PSO	0.03286	0.03945	0.04100	**0.00000**	0.15783
	EPSO	0.00108	0.00117	0.00033	**0.00010**	0.00125		EPSO	8.20397	8.59076	2.84563	3.16795	13.15980		EPSO	0.00591	**0.00000**	0.01445	**0.00000**	0.03945
	ABC	0.00117	**0.00010**	**0.00000**	0.00117	**0.00117**		ABC	**0.91220**	**0.15323**	**1.44928**	**0.01712**	**4.69119**		ABC	**0.00003**	**0.00000**	0.00011	**0.00000**	0.00050
	MABC	0.00113	**0.00010**	0.0002	0.00027	**0.00117**		MABC	44.68880	27.22880	29.94650	23.93810	112.734		MABC	0.00203	**0.00000**	0.00881	**0.00000**	0.03945
	MABC+ADVM	0.001186	0.00118	3.1e-05	0.001177	0.001349		MABC+ADVM	13.5615	5.57401	21.179	0.055279	78.1312		MABC+ADVM	38.1889	20.0484	44.4305	1.03001	182.301
	GBESTABC	0.00108	**0.00010**	0.00033	**0.00010**	0.00120		GBESTABC	1.70578	1.09851	1.85263	0.08547	6.11791		GBESTABC	0.00757	1.37E-05	0.01276	**0.00000**	0.03945
	GBESTABC2	0.00118	**0.00010**	1.13E-05	0.00117	0.00120		GBESTABC2	3.18224	2.5852	2.93224	0.35930	13.16040		GBESTABC2	0.00236	0.00015	0.00852	**0.00000**	0.03845
	ABC+A-DVM	**0.00105**	**0.00010**	0.00032	**0.00010**	**0.00117**		ABC+A-DVM	2.55989	1.04791	4.37552	0.02863	19.18300		ABC+A-DVM	0.00064	**0.00000**	0.00226	**0.00000**	0.01009
	GBESTABC+A-DVM	0.00113	**0.00010**	0.00024	**0.00010**	0.00125		GBESTABC+A-DVM	3.46204	1.48614	4.40623	0.04360	13.87970		GBESTABC+A-DVM	0.00618	**0.00000**	0.01435	**0.00000**	0.03945
	GBESTABC2+A-DVM	0.00118	**0.00010**	**0.00000**	0.00117	0.00120		GBESTABC2+A-DVM	4.55452	3.8697	3.73467	0.11879	14.8367		GBESTABC2+A-DVM	0.00062	0.00022	**0.00095**	**0.00000**	**0.00379**
	ABCXm1	0.011439	0.002754	0.010681	0.001389	0.028141		ABCXm1	72.0576	82.1424	28.3858	7.44934	109.364		ABCXm1	85.0402	57.9288	67.5181	26.3005	270.85
	ABCXm1+A-DVM	0.008005	0.002100	0.00942	0.001622	0.030850		ABCXm1+A-DVM	64.3322	77.9291	32.1162	2.19893	106.097		ABCXm1+A-DVM	62.608	44.1697	51.6055	15.8966	269.044
	ABCXm5	0.018339	0.015057	0.01484	0.0001	0.055938		ABCXm5	22.21	19.2556	9.54515	9.69884	47.4673		ABCXm5	59.9045	58.6329	14.0889	38.66390	102.5700
	ABCXm5+A-DVM	0.025878	0.014298	0.032853	0.00118	0.173876		ABCXm5+A-DVM	18.6672	14.0424	15.1627	5.32653	82.7050		ABCXm5+A-DVM	61.429	59.2038	15.6881	37.0757	100.947
Damavandi	DE	2.00000	2.00000	**0.00000**	2.00000	2.00000	Schwefel06	DE	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**	Zimmerman	DE	0.38522	0.69869	0.35633	**0.00000**	0.70133
	PSO	2.00000	2.00000	**0.00000**	2.00000	2.00000		PSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		PSO	715.1750	1300	663.34500	**0.00000**	1300.000
	EPSO	1.90000	2.00000	0.44721	**0.00000**	2.00000		EPSO	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		EPSO	0.17464	**0.00000**	0.31035	**0.00000**	0.69858
	ABC	2.00000	2.00000	**0.00000**	2.00000	2.00000		ABC	0.17159	0.10386	0.17989	0.00056	0.59950		ABC	0.00037	**0.00000**	0.00126	**0.00000**	0.00569
	MABC	2.00000	2.00000	**0.00000**	2.00000	2.00000		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		MABC	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**
	MABC+ADVM	1.61113	2.00000	0.778219	0.001555	2.00000		MABC+ADVM	0.123341	0.076095	0.180687	0.002949	0.767581		MABC+ADVM	0.000315	1.7e-05	0.001023	0.000000	0.005673
	GBESTABC	1.76059	2.00000	0.60220	0.00611	2.00000		GBESTABC	0.13196	0.11892	0.08783	0.01193	0.28331		GBESTABC	0.10626	0.00053	0.25554	**0.00000**	0.69923
	GBESTABC2	1.82772	2.00000	0.53654	0.02502	2.00000		GBESTABC2	0.13886	0.12197	0.10182	0.01223	0.42978		GBESTABC2	0.07062	0.00050	0.21487	5.96E-05	0.69904
	ABC+A-DVM	1.80056	2.00000	0.61387	0.00258	2.00000		ABC+A-DVM	0.11728	0.04898	0.17793	0.00272	0.72679		ABC+A-DVM	0.00041	3.27E-05	0.00097	**0.00000**	0.00325
	GBESTABC+A-DVM	1.81787	2.00000	0.56099	0.11335	2.00000		GBESTABC+A-DVM	0.09749	0.10197	0.04416	0.02492	0.16431		GBESTABC+A-DVM	0.10543	0.00035	0.25582	**0.00000**	0.69921
	GBESTABC2+A-DVM	**1.74073**	2.00000	0.63782	0.00028	2.00000		GBESTABC2+A-DVM	0.14695	0.09799	0.12129	0.02657	0.46299		GBESTABC2+A-DVM	0.03531	0.00020	0.15612	1.40E-05	0.69862
	ABCXm5	2.00000	2.00000	**0.00000**	2.00000	2.00000		ABCXm5	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABCXm5	0.232862	**0.00000**	0.329315	**0.00000**	0.698586
	ABCXm5+A-DVM	2.00000	2.00000	**0.00000**	2.00000	2.00000		ABCXm5+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABCXm5+A-DVM	0.349292	0.349292	0.349292	**0.00000**	0.698586
	ABCXm1	2.00000	2.00000	**0.00000**	2.00000	2.00000		ABCXm1	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABCXm1	0.256146	**0.00000**	0.336642	**0.00000**	0.698581
	ABCXm1+A-DVM	1.93333	2.00000	0.35901	**0.00000**	2.00000		ABCXm1+A-DVM	**0.00000**	**0.00000**	**0.00000**	**0.00000**	**0.00000**		ABCXm1+A-DVM	0.326004	**0.00000**	0.348513	**0.00000**	0.698581

Bukin06	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.4323e-11	0.209922	1.48463e-08	0.000219796	0.128748	1.82244e-10	0.168608	0.000463291	0.173632	5.37741e-10	1.4323e-11	1.4323e-11	0.138314	1.48463e-08
ABC-ES	-	1.50993e-11	5.13867e-07	3.87146e-10	1.50993e-11	6.23853e-05	1.47713e-11	4.0485e-10	1.50993e-11	2.22121e-07	0.0424998	0.018891	1.50993e-11	5.13867e-07
ABCx+A-DVM-m5	-	-	2.76643e-08	0.000340655	0.449998	1.30493e-10	0.444124	4.89476e-05	0.214482	9.27889e-10	1.50993e-11	1.50993e-11	0.181611	2.76643e-08
MABC	-	-	-	0.00251642	3.25914e-09	0.0328356	4.17254e-09	0.0025418	4.63014e-09	0.255296	2.5461e-08	1.00761e-08	2.28629e-09	0.18406
ABC+A-DVM	-	-	-	-	5.42735e-05	2.85674e-06	3.23538e-05	0.444024	0.000117432	4.43709e-05	2.66077e-10	8.46204e-11	0.000124434	0.00251642
ABCx-m5	-	-	-	-	-	9.7839e-11	0.46758	2.31892e-05	0.294726	3.05885e-10	1.50993e-11	1.50993e-11	0.255299	3.25914e-09
GBESTABC+A-DVM	-	-	-	-	-	-	1.05577e-10	3.14047e-06	1.18573e-10	0.135349	3.1414e-06	4.42055e-07	8.88454e-11	0.0328356
EPSO	-	-	-	-	-	-	-	3.52994e-05	0.479356	3.2109e-10	1.47713e-11	1.47713e-11	0.392179	4.17254e-09
ABC	-	-	-	-	-	-	-	-	2.04147e-05	7.01841e-05	1.73612e-10	9.77827e-11	1.38593e-05	0.0025418
ABCx-m1	-	-	-	-	-	-	-	-	-	4.44312e-10	1.50993e-11	1.50993e-11	0.378099	4.63014e-09
GBESTABC	-	-	-	-	-	-	-	-	-	-	2.15438e-08	2.09022e-09	4.87515e-10	0.255296
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.353086	1.50993e-11	2.5461e-08
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	1.50993e-11	1.00761e-08
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	2.28629e-09
Cola	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50898e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11
ABC-ES	-	0.264891	0.00278497	0.342161	0.19355	0.0481314	0.00348583	0.205955	0.000557128	0.0175683	0.185539	0.0511631	3.79957e-07	0.00278497
ABCx+A-DVM-m5	-	-	0.00473413	0.467596	0.380914	0.066727	0.00515681	0.310202	0.00102617	0.0362228	0.432497	0.0481314	5.9684e-07	0.00473413
MABC	-	-	-	0.0103404	0.0217918	0.218821	0.028728	0.0111801	0.038636	0.255299	0.0317663	0.185539	6.23853e-05	0.18406
ABC+A-DVM	-	-	-	-	0.38656	0.148636	0.00473364	0.432497	0.00151697	0.0789878	0.358594	0.103103	8.64516e-07	0.0103404
ABCx-m5	-	-	-	-	-	0.181611	0.00636544	0.432497	0.00201649	0.0789878	0.491154	0.155594	1.24566e-06	0.0217918
GBESTABC+A-DVM	-	-	-	-	-	-	0.0120775	0.103103	0.0103404	0.461721	0.201769	0.420901	2.98528e-05	0.218821
EPSO	-	-	-	-	-	-	-	0.00538076	0.380913	0.0188897	0.00955536	0.00814166	0.409372	0.028728
ABC	-	-	-	-	-	-	-	-	0.00102617	0.0648351	0.497051	0.111286	1.24566e-06	0.0111801
ABCx-m1	-	-	-	-	-	-	-	-	-	0.0242067	0.00291408	0.00720609	0.181611	0.038636
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.0978954	0.438317	7.03343e-05	0.255299
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.170144	5.09384e-06	0.0317663
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	1.13901e-05	0.185539
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	6.23853e-05
CrossLegTable	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.88668e-10	6.15459e-05	0.00988418	8.29814e-06	0.000401945	0.0738762	0.125063	4.85361e-07	0.000186338	0.181179	0.0507516	0.055547	0.000186338	0.00988418
ABC-ES	-	7.0328e-10	2.59815e-12	9.38541e-12	2.78175e-10	1.19757e-11	1.37041e-11	1.10665e-11	6.43181e-10	1.17704e-11	6.81594e-12	8.24062e-12	8.46924e-10	2.59815e-12
ABCx+A-DVM-m5	-	-	6.66123e-11	3.34893e-11	0.432483	2.41841e-09	8.32168e-09	2.57089e-11	0.0611635	4.60855e-09	5.24118e-10	7.24385e-10	0.0768193	6.66123e-11
MABC	-	-	-	2.00663e-05	8.45378e-10	0.175828	0.153162	2.20242e-07	2.60004e-12	0.0467708	0.146779	0.154137	2.60004e-12	0.18406
ABC+A-DVM	-	-	-	-	5.55389e-10	0.00258024	0.00551008	0.0612246	9.39154e-12	0.00014159	2.62516e-05	5.04706e-05	9.39154e-12	2.00663e-05
ABCx-m5	-	-	-	-	-	5.96732e-09	1.26859e-08	4.07016e-10	0.119918	9.61915e-09	2.85227e-09	3.41938e-09	0.132156	8.45378e-10
GBESTABC+A-DVM	-	-	-	-	-	-	0.308042	0.000411954	1.19833e-11	0.253375	0.396563	0.421904	1.19833e-11	0.175828
EPSO	-	-	-	-	-	-	-	0.0012239	5.50656e-11	0.48183	0.276969	0.288925	4.10246e-11	0.153162
ABC	-	-	-	-	-	-	-	-	1.10736e-11	1.06775e-05	8.00576e-07	1.62891e-06	1.10736e-11	2.20242e-07
ABCx-m1	-	-	-	-	-	-	-	-	-	1.1778e-11	6.82052e-12	8.24607e-12	0.159152	2.60004e-12
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.160971	0.176564	1.1778e-11	0.0467708
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.4831	6.82052e-12	0.146779
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	8.24607e-12	0.154137
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	2.60004e-12
CrownedCross	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.88774e-10	6.15459e-05	0.00313874	2.16704e-05	0.000401945	0.0308299	0.125063	1.24881e-05	0.000186338	0.225414	0.0131181	0.156157	0.000186267	0.00313874
ABC-ES	-	7.03648e-10	3.88432e-12	8.46104e-12	2.78328e-10	1.2895e-11	1.37128e-11	9.49637e-12	1.74856e-09	1.19987e-11	1.01877e-11	7.7844e-12	3.68616e-10	3.88432e-12
ABCx+A-DVM-m5	-	-	7.59201e-11	3.5606e-11	0.432483	1.76096e-09	8.32168e-09	3.66889e-11	0.135336	5.53236e-09	5.39866e-10	1.61048e-09	0.0225627	7.59201e-11
MABC	-	-	-	0.00272637	1.01263e-09	0.391053	0.108087	0.00110036	3.88432e-12	0.0149546	0.414888	0.00657923	3.87885e-12	0.18406
ABC+A-DVM	-	-	-	-	5.93316e-10	0.0257818	0.00865908	0.338518	8.46104e-12	8.60446e-05	0.0107089	7.74196e-06	8.44989e-12	0.00272637
ABCx-m5	-	-	-	-	-	7.12191e-09	1.26859e-08	6.34878e-10	0.27961	1.49781e-08	3.31439e-09	6.25234e-09	0.0374054	1.01263e-09
GBESTABC+A-DVM	-	-	-	-	-	-	0.273051	0.0170544	1.2895e-11	0.119454	0.432415	0.141106	1.28786e-11	0.391053
EPSO	-	-	-	-	-	-	-	0.0072814	6.07163e-11	0.375122	0.173769	0.5	8.94474e-11	0.108087
ABC	-	-	-	-	-	-	-	-	9.49637e-12	4.65081e-05	0.00577667	3.90672e-06	9.48398e-12	0.00110036
ABCx-m1	-	-	-	-	-	-	-	-	-	1.19987e-11	1.01877e-11	7.7844e-12	0.00636482	3.88432e-12
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.0472774	0.351055	1.19833e-11	0.0149546
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.0482569	1.01745e-11	0.414888
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	7.77407e-12	0.00657923
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	3.87885e-12

Table 3.9: Pairwise U-test analysis from Bukin06 to Crowned Cross functions

Damavandi	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	5.82529e-14	0.18406	0.00551754	0.00551754	0.18406	0.0804011	0.0054458	0.0804011	0.18406	0.166855	0.0407615	0.0407615	0.166855	0.00551754
ABC-ES	-	5.82529e-14	7.92983e-13	7.92983e-13	5.82529e-14	1.63123e-13	1.23252e-09	1.63123e-13	5.82529e-14	9.97923e-14	2.55617e-13	2.55617e-13	4.01815e-13	7.92983e-13
ABCx+A-DVM-m5	-	-	0.00551754	0.00551754	0.18406	0.0804011	0.0054458	0.0804011	0.18406	0.166855	0.0407615	0.0407615	0.166855	0.00551754
MABC	-	-	-	0.453671	0.00551754	0.0903332	0.355428	0.0682646	0.00551754	0.0289726	0.119396	0.199101	0.0289726	0.18406
ABC+A-DVM	-	-	-	-	0.00551754	0.0903332	0.355428	0.0650302	0.00551754	0.0289726	0.119396	0.199101	0.0289726	0.453671
ABCx-m5	-	-	-	-	-	0.0804011	0.0054458	0.0804011	0.18406	0.166855	0.0407615	0.0407615	0.166855	0.00551754
GBESTABC+A-DVM	-	-	-	-	-	-	0.0503713	0.47955	0.0804011	0.298595	0.361414	0.29469	0.298595	0.0903332
EPSO	-	-	-	-	-	-	-	0.0503713	0.0054458	0.0197735	0.101188	0.101188	0.0238162	0.355428
ABC	-	-	-	-	-	-	-	-	0.0804011	0.298595	0.349924	0.29469	0.298595	0.0682646
ABCx-m1	-	-	-	-	-	-	-	-	-	0.166855	0.0407615	0.0407615	0.166855	0.00551754
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.169209	0.152523	0.5	0.0289726
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.454781	0.169209	0.119396
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	0.169209	0.199101
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	0.0289726

DeVilliersGlasser02	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.50993e-11	1.50993e-11	1.50993e-11	6.0589e-13	1.50993e-11	8.60126e-13	1.66919e-11	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	1.50993e-11	1.50993e-11
ABC-ES	-	4.87775e-10	1.50993e-11	6.0589e-13	2.09984e-10	8.60126e-13	1.50993e-11	6.0589e-13	0.0151587	6.0589e-13	6.0589e-13	6.0589e-13	0.152088	1.50993e-11
ABCx+A-DVM-m5	-	-	1.50993e-11	6.0589e-13	0.245891	8.60126e-13	1.50993e-11	6.0589e-13	3.82939e-05	6.0589e-13	6.0589e-13	6.0589e-13	2.05888e-06	1.50993e-11
MABC	-	-	-	6.0589e-13	1.50993e-11	8.60126e-13	5.46835e-11	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	2.48758e-11	0.18406
ABC+A-DVM	-	-	-	-	6.0589e-13	0.166855	6.0589e-13	0.18406	6.0589e-13	0.18406	0.18406	0.18406	6.0589e-13	6.0589e-13
ABCx-m5	-	-	-	-	-	8.60126e-13	1.50993e-11	6.0589e-13	2.6325e-05	6.0589e-13	6.0589e-13	6.0589e-13	1.66208e-06	1.50993e-11
GBESTABC+A-DVM	-	-	-	-	-	-	8.60126e-13	0.166855	1.2055e-12	0.166855	0.166855	0.166855	1.07747e-12	8.60126e-13
EPSO	-	-	-	-	-	-	-	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	1.50993e-11	5.46835e-11
ABC	-	-	-	-	-	-	-	-	6.0589e-13	0.18406	0.18406	0.18406	6.0589e-13	6.0589e-13
ABCx-m1	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13	6.0589e-13	0.189518	1.50993e-11
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	6.0589e-13	6.0589e-13
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.18406	6.0589e-13	6.0589e-13
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	2.48758e-11

Griewank	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.19833e-11	1.50993e-11	1.50709e-11	6.0589e-13	1.50898e-11	6.0589e-13	1.18284e-12	8.60126e-13	1.50993e-11	8.60126e-13	6.0589e-13	1.18192e-12	1.50898e-11	1.50709e-11
ABC-ES	-	5.81883e-10	0.00772719	1.10591e-06	4.39882e-10	1.10591e-06	0.000100603	1.25899e-05	1.05827e-10	3.93803e-06	1.10591e-06	7.56738e-06	2.50191e-10	0.00772719
ABCx+A-DVM-m5	-	-	0.000601919	6.0589e-13	0.41801	6.0589e-13	6.19215e-10	2.28091e-11	1.73883e-05	8.60126e-13	6.0589e-13	1.18192e-12	5.4519e-06	0.000601919
MABC	-	-	-	8.27367e-12	0.000528437	8.27367e-12	5.04433e-09	2.27682e-10	1.38985e-07	3.32294e-11	8.27367e-12	2.55281e-11	3.01539e-07	0.18406
ABC+A-DVM	-	-	-	-	6.05396e-13	0.18406	0.0804011	0.166855	6.0589e-13	0.166855	0.18406	0.080371	6.05396e-13	8.27367e-12
ABCx-m5	-	-	-	-	-	6.05396e-13	6.18852e-10	2.27933e-11	2.89174e-05	8.59442e-13	6.05396e-13	1.181e-12	7.14526e-06	0.000528437
GBESTABC+A-DVM	-	-	-	-	-	-	0.0804011	0.166855	6.0589e-13	0.166855	0.18406	0.080371	6.05396e-13	8.27367e-12
EPSO	-	-	-	-	-	-	-	0.272013	3.44112e-10	0.272013	0.0804011	0.479548	3.79508e-10	5.04433e-09
ABC	-	-	-	-	-	-	-	-	3.27579e-12	0.5	0.166855	0.298577	4.54799e-12	2.27682e-10
ABCx-m1	-	-	-	-	-	-	-	-	-	8.60126e-13	6.0589e-13	1.18192e-12	0.353084	1.38985e-07
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.166855	0.298577	8.59442e-13	3.32294e-11
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.080371	6.05396e-13	8.27367e-12
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	1.181e-12	2.55281e-11
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	3.01539e-07

Rastrigin	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.50993e-11	1.50993e-11	1.50993e-11	6.0589e-13	1.50993e-11	8.60126e-13	1.66919e-11	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	1.50993e-11	1.50993e-11
ABC-ES	-	4.87775e-10	1.50993e-11	6.0589e-13	2.09984e-10	8.60126e-13	1.50993e-11	6.0589e-13	0.0151587	6.0589e-13	6.0589e-13	6.0589e-13	0.152088	1.50993e-11
ABCx+A-DVM-m5	-	-	1.50993e-11	6.0589e-13	0.245891	8.60126e-13	1.50993e-11	6.0589e-13	3.82939e-05	6.0589e-13	6.0589e-13	6.0589e-13	2.05888e-06	1.50993e-11
MABC	-	-	-	6.0589e-13	1.50993e-11	8.60126e-13	5.46835e-11	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	2.48758e-11	0.18406
ABC+A-DVM	-	-	-	-	6.0589e-13	0.166855	6.0589e-13	0.18406	6.0589e-13	0.18406	0.18406	0.18406	6.0589e-13	6.0589e-13
ABCx-m5	-	-	-	-	-	8.60126e-13	1.50993e-11	6.0589e-13	2.6325e-05	6.0589e-13	6.0589e-13	6.0589e-13	1.66208e-06	1.50993e-11
GBESTABC+A-DVM	-	-	-	-	-	-	8.60126e-13	0.166855	1.2055e-12	0.166855	0.166855	0.166855	1.07747e-12	8.60126e-13
EPSO	-	-	-	-	-	-	-	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	1.50993e-11	5.46835e-11
ABC	-	-	-	-	-	-	-	-	6.0589e-13	0.18406	0.18406	0.18406	6.0589e-13	6.0589e-13
ABCx-m1	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13	6.0589e-13	0.189518	1.50993e-11
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	6.0589e-13	6.0589e-13
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.18406	6.0589e-13	6.0589e-13
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	2.48758e-11

Table 3.10: Pairwise U-test analysis from Damavandi to Rastrigin functions

Rosenbrock	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.50993e-11	1.50993e-11	1.50993e-11	6.0589e-13	1.50993e-11	8.60126e-13	1.66919e-11	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	1.50993e-11	1.50993e-11
ABC-ES	-	4.87775e-10	1.50993e-11	6.0589e-13	2.09984e-10	8.60126e-13	1.50993e-11	6.0589e-13	0.0151587	6.0589e-13	6.0589e-13	6.0589e-13	0.152088	1.50993e-11
ABCx+A-DVM-m5	-	-	1.50993e-11	6.0589e-13	0.245891	8.60126e-13	1.50993e-11	6.0589e-13	3.82939e-05	6.0589e-13	6.0589e-13	6.0589e-13	2.05888e-06	1.50993e-11
MABC	-	-	-	6.0589e-13	1.50993e-11	8.60126e-13	5.46835e-11	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	2.48758e-11	0.18406
ABC+A-DVM	-	-	-	-	6.0589e-13	0.166855	6.0589e-13	0.18406	6.0589e-13	0.18406	0.18406	0.18406	6.0589e-13	6.0589e-13
ABCx-m5	-	-	-	-	-	8.60126e-13	1.50993e-11	6.0589e-13	2.6325e-05	6.0589e-13	6.0589e-13	6.0589e-13	1.66208e-06	1.50993e-11
GBESTABC+A-DVM	-	-	-	-	-	-	8.60126e-13	0.166855	1.2055e-12	0.166855	0.166855	0.166855	1.07747e-12	8.60126e-13
EPSO	-	-	-	-	-	-	-	6.0589e-13	1.50993e-11	6.0589e-13	6.0589e-13	6.0589e-13	1.50993e-11	5.46835e-11
ABC	-	-	-	-	-	-	-	-	6.0589e-13	0.18406	0.18406	0.18406	6.0589e-13	6.0589e-13
ABCx-m1	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13	6.0589e-13	0.189518	1.50993e-11
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	6.0589e-13	6.0589e-13
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.18406	6.0589e-13	6.0589e-13
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	2.48758e-11

Schwefel06	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	0.0013875	6.90317e-12	6.0589e-13	6.0589e-13	2.1558e-11	8.27367e-12	0.18406	6.0589e-13	0.18406	9.6525e-11	6.05396e-13	6.0589e-13	0.18406	6.0589e-13
ABC-ES	-	1.46217e-07	4.66114e-12	4.66114e-12	1.83084e-06	1.75567e-10	0.0013875	4.66114e-12	0.0013875	7.12005e-09	4.65791e-12	4.66114e-12	0.0013875	4.66114e-12
ABCx+A-DVM-m5	-	-	1.29772e-11	1.29772e-11	0.120463	8.41872e-08	6.90317e-12	1.29772e-11	6.90317e-12	1.50954e-05	1.2969e-11	1.29772e-11	6.90317e-12	1.29772e-11
MABC	-	-	-	0.100474	1.17328e-11	1.50709e-11	6.0589e-13	0.0496288	6.0589e-13	1.49767e-11	3.05885e-10	1.59837e-09	6.0589e-13	0.18406
ABC+A-DVM	-	-	-	-	1.17328e-11	1.50709e-11	6.0589e-13	0.294726	6.0589e-13	1.6557e-11	1.53985e-08	5.78327e-08	6.0589e-13	0.100474
ABCx-m5	-	-	-	-	-	1.57018e-08	2.1558e-11	1.17328e-11	2.1558e-11	3.47585e-06	1.17253e-11	1.17328e-11	2.1558e-11	1.17328e-11
GBESTABC+A-DVM	-	-	-	-	-	-	8.27367e-12	1.84143e-11	8.27367e-12	0.160786	2.41277e-10	2.248e-11	8.27367e-12	1.50709e-11
EPSO	-	-	-	-	-	-	-	6.0589e-13	0.18406	9.6525e-11	6.05396e-13	6.0589e-13	0.18406	6.0589e-13
ABC	-	-	-	-	-	-	-	-	6.0589e-13	2.7253e-11	1.01375e-07	7.45902e-07	6.0589e-13	0.0496288
ABCx-m1	-	-	-	-	-	-	-	-	-	9.6525e-11	6.05396e-13	6.0589e-13	0.18406	6.0589e-13
GBESTABC	-	-	-	-	-	-	-	-	-	-	5.82549e-10	1.0692e-10	9.6525e-11	1.49767e-11
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.17966	6.05396e-13	3.05885e-10
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	6.0589e-13	1.59837e-09
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	6.0589e-13

SineEnvelope	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	3.69454e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11
ABC-ES	-	3.06052e-10	1.07702e-06	1.66919e-11	3.06052e-10	1.50993e-11	1.50993e-11	1.50993e-11	3.26307e-07	1.50993e-11	1.50993e-11	1.50993e-11	2.4713e-05	1.07702e-06
ABCx+A-DVM-m5	-	-	2.03858e-11	1.09737e-08	0.380914	5.46835e-11	1.50993e-11	1.11363e-09	0.461721	8.06613e-11	4.07637e-11	1.30493e-10	0.12594	2.03858e-11
MABC	-	-	-	1.50993e-11	2.48758e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.57944e-10	1.50993e-11	1.50993e-11	1.50993e-11	1.91245e-09	0.18406
ABC+A-DVM	-	-	-	-	7.79038e-09	0.00348622	1.50993e-11	0.369699	5.45343e-06	0.00131214	0.210193	0.173914	1.09794e-07	1.50993e-11
ABCx-m5	-	-	-	-	-	4.95931e-11	1.50993e-11	9.28367e-10	0.479366	5.46835e-11	5.46835e-11	1.57944e-10	0.132163	2.48758e-11
GBESTABC+A-DVM	-	-	-	-	-	-	1.50993e-11	0.00434219	2.34282e-08	0.467596	0.00348622	0.0362228	4.87775e-10	1.50993e-11
EPSO	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11
ABC	-	-	-	-	-	-	-	-	3.36811e-06	0.00211296	0.342161	0.274663	2.5461e-08	1.50993e-11
ABCx-m1	-	-	-	-	-	-	-	-	-	1.19487e-08	5.13867e-07	3.52149e-07	0.159152	1.57944e-10
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.00159148	0.0146027	5.35089e-10	1.50993e-11
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.420901	3.25914e-09	1.50993e-11
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	6.55552e-09	1.50993e-11
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	1.91245e-09

Trefethen	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.68597e-07	4.4196e-06	1.68597e-07	1.68597e-07	0.000911263	0.0473405	0.000372173	1.68597e-07	0.0207284	0.0280822	1.68597e-07	1.68597e-07	0.00163145	1.68597e-07
ABC-ES	-	0.0407021	0.18406	0.18406	0.000669594	3.19329e-05	0.00136379	0.18406	2.23578e-06	3.17451e-05	0.18406	0.18406	0.000678539	0.18406
ABCx+A-DVM-m5	-	-	0.0407021	0.0407021	0.0246025	0.00104272	0.0471179	0.0407021	0.000149753	0.00135916	0.0407021	0.0407021	0.0213792	0.0407021
MABC	-	-	-	0.18406	0.000669594	3.19329e-05	0.00136379	0.18406	2.23578e-06	3.17451e-05	0.18406	0.18406	0.000678539	0.18406
ABC+A-DVM	-	-	-	-	0.000669594	3.19329e-05	0.00136379	0.18406	2.23578e-06	3.17451e-05	0.18406	0.18406	0.000678539	0.18406
ABCx-m5	-	-	-	-	-	0.0749114	0.374724	0.000669594	0.0438242	0.109604	0.000669594	0.000669594	0.445246	0.000669594
GBESTABC+A-DVM	-	-	-	-	-	-	0.0450536	3.19329e-05	0.474152	0.411891	3.19329e-05	3.19329e-05	0.0982305	3.19329e-05
EPSO	-	-	-	-	-	-	-	0.00136379	0.0184559	0.0623302	0.00136379	0.00136379	0.326643	0.00136379
ABC	-	-	-	-	-	-	-	-	2.23578e-06	3.17451e-05	0.18406	0.18406	0.000678539	0.18406
ABCx-m1	-	-	-	-	-	-	-	-	-	0.375015	2.23578e-06	2.23578e-06	0.0753994	2.23578e-06
GBESTABC	-	-	-	-	-	-	-	-	-	-	3.17451e-05	3.17451e-05	0.144437	3.17451e-05
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.18406	0.000678539	0.18406
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	0.000678539	0.18406
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	0.000678539

Table 3.11: Pairwise U-test analysis from Rosenbrock to Trefethen functions

Whitley	ABC-ES	ABCx+A-DVM-m5	MABC	ABC+A-DVM	ABCx-m5	GBESTABC+A-DVM	EPSO	ABC	ABCx-m1	GBESTABC	GBESTABC2+A-DVM	GBESTABC2	ABCx+A-DVM-m1	MABC+A-DVM
PSO	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11
ABC-ES	-	7.14708e-09	2.34282e-08	4.24239e-09	2.09125e-09	2.04198e-05	1.50993e-11	2.53616e-10	0.000405999	1.59837e-09	3.88627e-09	4.42055e-07	3.1414e-06	2.34282e-08
ABCx+A-DVM-m5	-	-	2.98528e-05	4.60563e-05	0.397923	0.000199405	1.50993e-11	2.17654e-05	0.467596	2.21025e-06	1.90263e-07	0.000119424	0.00720609	2.98528e-05
MABC	-	-	-	0.299845	2.4713e-05	0.205955	1.50993e-11	0.331367	7.92305e-05	0.105781	0.201769	0.116994	0.00515734	0.18406
ABC+A-DVM	-	-	-	-	6.62476e-05	0.369699	1.50993e-11	0.485258	1.68407e-05	0.205955	0.497051	0.210193	0.000249091	0.299845
ABCx-m5	-	-	-	-	-	0.000199405	1.50993e-11	3.17802e-05	0.409373	2.73102e-06	1.62777e-07	0.000126529	0.00538131	2.4713e-05
GBESTABC+A-DVM	-	-	-	-	-	-	1.50993e-11	0.39223	4.89586e-05	0.315438	0.218821	0.245891	0.000168395	0.205955
EPSO	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11
ABC	-	-	-	-	-	-	-	-	1.68407e-05	0.22321	0.485258	0.236673	0.000883282	0.331367
ABCx-m1	-	-	-	-	-	-	-	-	-	2.21025e-06	2.59284e-07	1.29868e-05	0.0328356	7.92305e-05
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.145236	0.479366	2.98528e-05	0.105781
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.0953652	7.14918e-06	0.201769
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	3.59939e-05	0.116994
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00515734
XinSheYang06	**ABC-ES**	**ABCx+A-DVM-m5**	**MABC**	**ABC+A-DVM**	**ABCx-m5**	**GBESTABC+A-DVM**	**EPSO**	**ABC**	**ABCx-m1**	**GBESTABC**	**GBESTABC2+A-DVM**	**GBESTABC2**	**ABCx+A-DVM-m1**	**MABC+A-DVM**
PSO	0.0407021	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
ABC-ES	-	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021	0.0407021
ABCx+A-DVM-m5	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
MABC	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
ABC+A-DVM	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
ABCx-m5	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
GBESTABC+A-DVM	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
EPSO	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
ABC	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406
ABCx-m1	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	0.18406
Zimmerman	**ABC-ES**	**ABCx+A-DVM-m5**	**MABC**	**ABC+A-DVM**	**ABCx-m5**	**GBESTABC+A-DVM**	**EPSO**	**ABC**	**ABCx-m1**	**GBESTABC**	**GBESTABC2+A-DVM**	**GBESTABC2**	**ABCx+A-DVM-m1**	**MABC+A-DVM**
PSO	4.56246e-06	0.00191853	0.00145742	0.000903768	0.000513593	0.00646495	0.000540484	0.000903629	3.68175e-05	0.00340867	0.00765646	0.00866069	0.000138162	0.00145742
ABC-ES	-	0.000804015	0.000273693	0.000898623	0.0053294	4.97132e-06	0.0314519	0.000898472	0.218961	3.50841e-05	1.51208e-06	9.64117e-07	0.0840264	0.000273693
ABCx+A-DVM-m5	-	-	0.267523	0.204729	0.252784	0.0178275	0.0427109	0.211192	0.00320394	0.0260143	0.0955879	0.063022	0.0120503	0.267523
MABC	-	-	-	0.34173	0.104913	0.0273778	0.192421	0.389072	0.106618	0.0348831	0.0217261	0.00258444	0.490923	0.18406
ABC+A-DVM	-	-	-	-	0.152641	0.0291895	0.138743	0.467464	0.167898	0.0351092	0.0833004	0.00778878	0.394688	0.34173
ABCx-m5	-	-	-	-	-	0.00192754	0.367506	0.142265	0.0583064	0.00441018	0.00110716	0.000649672	0.179864	0.104913
GBESTABC+A-DVM	-	-	-	-	-	-	0.0183738	0.0296956	0.000427548	0.400006	0.40357	0.426642	0.00336902	0.0273778
EPSO	-	-	-	-	-	-	-	0.138739	0.0863635	0.0369074	0.186093	0.131925	0.268265	0.192421
ABC	-	-	-	-	-	-	-	-	0.167895	0.0351072	0.0756454	0.00633942	0.394687	0.389072
ABCx-m1	-	-	-	-	-	-	-	-	-	0.00162872	0.00258724	0.00152962	0.220545	0.106618
GBESTABC	-	-	-	-	-	-	-	-	-	-	0.423662	0.449913	0.00883817	0.0348831
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.107119	0.0440778	0.0217261
GBESTABC2	-	-	-	-	-	-	-	-	-	-	-	-	0.0277135	0.00258444
ABCx+A-DVM-m1	-	-	-	-	-	-	-	-	-	-	-	-	-	0.490923

Table 3.12: Pairwise U-test analysis from Whitley to Zimmerman functions

Griewank	GBESTABC	GBESTABC+A-DVM	ABC+A-DVM	ABC	ABCX-m1+A-DVM	ABCX-m5+A-DVM	MABC+A-DVM	GBESTABC2+A-DVM	MABC	ABCx-m1	ABCx-m5	ABC-ES
GBESTABC2	3.69014e-10	1.0772e-10	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.170144	1.50993e-11	1.50993e-11	1.50993e-11	0.0811875
GBESTABC	-	0.409373	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	6.43519e-10	1.50993e-11	1.50993e-11	1.50993e-11	0.0169371
GBESTABC+A-DVM	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	4.05068e-10	1.50993e-11	1.50993e-11	1.50993e-11	0.0233779
ABC+A-DVM	-	-	-	0.122907	1.50993e-11	5.52886e-05	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	7.92305e-05	0.236673
ABC	-	-	-	-	1.50993e-11	0.00254211	1.50993e-11	1.84486e-11	1.50993e-11	1.50993e-11	0.00610597	0.210193
ABCX-m1+A-DVM	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.497051	1.50993e-11	1.50993e-11
ABCX-m5+A-DVM	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.39223	0.122907
MABC+A-DVM	-	-	-	-	-	-	-	1.50993e-11	0.18406	1.50993e-11	1.50993e-11	0.00201649
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	0.0928834
MABC	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	0.00201649
ABCx-m1	-	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	0.132163
Rastrigin	GBESTABC	GBESTABC+A-DVM	ABC+A-DVM	ABC	ABCX-m1+A-DVM	ABCX-m5+A-DVM	MABC+A-DVM	GBESTABC2+A-DVM	MABC	ABCx-m1	ABCx-m5	ABC-ES
GBESTABC2	8.64516e-07	1.24566e-06	2.22202e-07	8.64516e-07	1.50993e-11	1.50993e-11	1.50993e-11	0.085725	1.50993e-11	1.50993e-11	1.50993e-11	0.491154
GBESTABC	-	0.38656	3.36098e-10	2.30796e-10	1.50993e-11	1.50993e-11	1.50993e-11	7.79038e-09	1.50993e-11	1.50993e-11	1.50993e-11	0.426691
GBESTABC+A-DVM	-	-	1.43579e-10	1.0772e-10	1.50993e-11	1.50993e-11	1.50993e-11	7.79038e-09	1.50993e-11	1.50993e-11	1.50993e-11	0.38656
ABC+A-DVM	-	-	-	0.241258	1.50993e-11	4.4455e-10	1.50993e-11	5.5386e-07	1.50993e-11	1.50993e-11	2.48758e-11	0.250572
ABC	-	-	-	-	1.50993e-11	1.11363e-09	1.50993e-11	5.45343e-06	1.50993e-11	1.50993e-11	3.34776e-11	0.260072
ABCX-m1+A-DVM	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.19355	1.50993e-11	1.50993e-11
ABCX-m5+A-DVM	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.0362228	0.0481314
MABC+A-DVM	-	-	-	-	-	-	-	1.50993e-11	0.18406	1.50993e-11	1.50993e-11	1.79617e-05
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	0.479366
MABC	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.79617e-05
ABCx-m1	-	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	0.0374135
Rosenbrock	GBESTABC	GBESTABC+A-DVM	ABC+A-DVM	ABC	ABCX-m1+A-DVM	ABCX-m5+A-DVM	MABC+A-DVM	GBESTABC2+A-DVM	MABC	ABCx-m1	ABCx-m5	ABC-ES
GBESTABC2	0.000126529	0.000119424	4.87775e-10	2.09984e-10	1.50993e-11	1.50993e-11	1.50993e-11	0.369699	1.50993e-11	1.50993e-11	1.50993e-11	0.0225731
GBESTABC	-	0.264891	4.07637e-11	6.02834e-11	1.50993e-11	1.50993e-11	1.50993e-11	9.4581e-05	1.50993e-11	1.50993e-11	1.50993e-11	0.000976338
GBESTABC+A-DVM	-	-	4.07637e-11	4.95931e-11	1.50993e-11	1.50993e-11	1.50993e-11	5.8736e-05	1.50993e-11	1.50993e-11	1.50993e-11	0.000758898
ABC+A-DVM	-	-	-	0.353086	1.50993e-11	0.000384864	1.84486e-11	5.35089e-10	1.84486e-11	1.50993e-11	0.000150294	0.250572
ABC	-	-	-	-	1.50993e-11	0.000839878	1.84486e-11	1.43579e-10	1.84486e-11	1.50993e-11	0.000405999	0.255299
ABCX-m1+A-DVM	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.030726	1.50993e-11	1.50993e-11
ABCX-m5+A-DVM	-	-	-	-	-	-	2.74703e-11	1.50993e-11	2.74703e-11	1.50993e-11	0.449998	0.135353
MABC+A-DVM	-	-	-	-	-	-	-	1.50993e-11	0.18406	1.50993e-11	2.25216e-11	0.00183545
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	0.0225731
MABC	-	-	-	-	-	-	-	-	-	1.50993e-11	2.25216e-11	0.00183545
ABCx-m1	-	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	0.122907
SineEnvelope	GBESTABC	GBESTABC+A-DVM	ABC+A-DVM	ABC	ABCX-m1+A-DVM	ABCX-m5+A-DVM	MABC+A-DVM	GBESTABC2+A-DVM	MABC	ABCx-m1	ABCx-m5	ABC-ES
GBESTABC2	0.00211296	5.20328e-05	0.369699	0.152088	1.50993e-11	1.54053e-08	2.25216e-11	0.364133	2.25216e-11	1.50993e-11	4.176e-08	0.444151
GBESTABC	-	0.19355	0.00201649	0.000451535	1.50993e-11	4.05068e-10	1.84486e-11	0.0125506	1.84486e-11	1.50993e-11	5.86868e-10	0.409373
GBESTABC+A-DVM	-	-	0.000119424	3.17802e-05	1.50993e-11	1.0772e-10	1.50993e-11	0.00092874	1.50993e-11	1.50993e-11	1.0772e-10	0.353086
ABC+A-DVM	-	-	-	0.28461	1.50993e-11	5.9684e-07	3.69454e-11	0.264891	3.69454e-11	1.50993e-11	9.30424e-07	0.426691
ABC	-	-	-	-	1.50993e-11	1.33921e-06	4.07637e-11	0.138595	4.07637e-11	1.50993e-11	4.75697e-06	0.364133
ABCX-m1+A-DVM	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.0726596	1.50993e-11	1.50993e-11
ABCX-m5+A-DVM	-	-	-	-	-	-	1.82295e-08	9.36549e-08	1.82295e-08	1.50993e-11	0.449998	0.0648351
MABC+A-DVM	-	-	-	-	-	-	-	2.74703e-11	0.18406	1.50993e-11	5.5117e-09	3.36811e-06
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	2.74703e-11	1.50993e-11	2.79995e-07	0.444151
MABC	-	-	-	-	-	-	-	-	-	1.50993e-11	5.5117e-09	3.36811e-06
ABCx-m1	-	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	0.0629851

Table 3.13: Pairwise U-test analysis of the Griewank, Rosenbrock, Rastrigin and SineEnvelope functions

	GBESTABC	GBESTABC+A-DVM	ABC+A-DVM	ABC	ABCX-m1+A-DVM	ABCX-m5+A-DVM	MABC+A-DVM	GBESTABC2+A-DVM	MABC	ABCx-m1	ABCx-m5	ABC-ES
GBESTABC2	0.000126529	3.59939e-05	2.78633e-10	2.09984e-10	1.50993e-11	1.50993e-11	1.50993e-11	0.38656	1.50993e-11	1.50993e-11	1.50993e-11	0.0250601
GBESTABC	-	0.342161	2.25216e-11	1.0772e-10	1.50993e-11	1.50993e-11	1.50993e-11	5.83718e-06	1.50993e-11	1.50993e-11	1.50993e-11	0.00750707
GBESTABC+A-DVM	-	-	1.50993e-11	8.88454e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.00115e-06	1.50993e-11	1.50993e-11	1.50993e-11	0.00561388
ABC+A-DVM	-	-	-	0.364133	1.50993e-11	6.24662e-06	1.66919e-11	2.53616e-10	1.66919e-11	1.50993e-11	0.000345626	0.22321
ABC	-	-	-	-	1.50993e-11	1.91533e-05	2.03858e-11	3.36098e-10	2.03858e-11	1.50993e-11	0.000428206	0.289647
ABCX-m1+A-DVM	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	0.00847744	1.50993e-11	1.50993e-11
ABCX-m5+A-DVM	-	-	-	-	-	-	2.48758e-11	1.50993e-11	2.48758e-11	1.50993e-11	0.108508	0.114115
MABC+A-DVM	-	-	-	-	-	-	-	1.50993e-11	0.18406	1.50993e-11	2.03858e-11	0.00561388
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11	1.50993e-11	0.029714
MABC	-	-	-	-	-	-	-	-	-	1.50993e-11	2.03858e-11	0.00561388
ABCx-m1	-	-	-	-	-	-	-	-	-	-	1.50993e-11	1.50993e-11
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	0.122907
Zimmerman	GBESTABC	GBESTABC+A-DVM	ABC+A-DVM	ABC	ABCX-m1+A-DVM	ABCX-m5+A-DVM	MABC+A-DVM	GBESTABC2+A-DVM	MABC	ABCx-m1	ABCx-m5	ABC-ES
GBESTABC2	6.89549e-05	2.92589e-05	0.189518	0.189518	0.181611	0.129026	0.132163	0.299845	0.132163	0.326022	0.132163	0.353086
GBESTABC	-	0.315678	0.497012	0.497012	0.000850874	0.0125714	0.0367378	0.00178867	0.0367378	0.00687226	0.0125714	0.329299
GBESTABC+A-DVM	-	-	0.328816	0.328816	0.000352362	0.00544148	0.0167606	0.000742985	0.0167606	0.00291886	0.00544148	0.193839
ABC+A-DVM	-	-	-	0.0904498	0.331367	0.0157328	8.88454e-11	0.189518	8.88454e-11	0.038636	0.0398908	0.0629851
ABC	-	-	-	-	0.331367	0.0140643	3.03288e-11	0.189518	3.03288e-11	0.038636	0.038636	0.279615
ABCX-m1+A-DVM	-	-	-	-	-	0.0953652	0.0182194	0.135353	0.0182194	0.111286	0.085725	0.197634
ABCX-m5+A-DVM	-	-	-	-	-	-	0.000345626	0.0210334	0.000345626	0.0049417	0.485258	0.00691581
MABC+A-DVM	-	-	-	-	-	-	-	0.00781906	0.18406	0.00107832	0.000618092	9.36549e-08
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	0.00781906	0.409373	0.0175683	0.103103
MABC	-	-	-	-	-	-	-	-	-	0.00107832	0.000618092	9.36549e-08
ABCx-m1	-	-	-	-	-	-	-	-	-	-	0.00380853	0.0182194
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	0.0169371

Table 3.14: Pairwise U-test analysis of the Whitley and Zimmerman functions

ABC algorithm was observed have better results, holding statistical significance against all but the ABCX-m1+A-DVM algorithm.

First, in the Griewank instance, there was no statistical relevance between the original and the A-DVM counterparts, where it can be understood that although the ABCX-m1 was more robust, i.e., lower mean, than the original, it was not substantially better. The same cannot be said about the Rosenbrock instance. A trend was observed in the improvement of the MABC+A-DVM compared to the MABC ($p < 0.1$) while the ABCx-m1+ADVM held statistical difference compared to its non A-DVM version, corroborating the evidence of its underperformance. On the other hand, the inclusion of the A-DVM in the ABCX-m1 incurred in a substantial improvement backed up by statistical significance ($p < 0.05$).

For the three last instances, once again, there is a minor statistical evidence that the robustness of the ABCX-m1+A-DVM is tighter than the ABCX-m1 ($0.05 < p \leq 0.1$). In the Whitley function, the inclusion of the A-DVM brought an improvement to the performance of the ABCX-m1 and at the same time, worsened the robustness of the ABCX-m5, both assertions are backed up by statistical relevance ($p < 0.05$). Lastly, in the Zimmerman instance where the ABC and the ABC+A-DVM had the best performance, there is little statistical evidence whether the addition of the A-DVM contributed to the improvement of the results ($0.05 < p < 0.1$).

Summarizing, it was observed that a slight improvement with the inclusion of the A-DVM to the improvement of the robustness of the algorithms in the large-scale instances. It is important to highlight some few cases where the A-DVM was able to take advantage of the features of the base algorithm and even improve its robustness. One possible reason would be that since only one decision variable is changed at a time via the A-DVM, convergence of solutions to accumulation points is delayed or even fully compromised. A possible direction to further investigate would be the generalization of the A-DVM mechanism to select multiple solutions variables at a time, specifically for large-scale instances such as the ones in this experiment.

3.3.3 Testing the A-DVM against other derivative-free algorithms

The A-DVM based algorithms are now tested against other derivative free algorithms following the guidelines of Gavana [55]. We chose the same 15 hardest instances from the global optimization benchmark using a strenuous amount of iterations to test the speed of convergence of the A-DVM based algorithms. The ABC+A-DVM, ABCX-M1+A-DVM, GBESTABC2+A-DVM and MABC+A-DVM were chosen to based on the results of Sections 3.3.1 and 3.3.2. The derivative-free techniques which the A-DVM algorithms are compared against are the following: the Adaptive memory Programming for Global Optimization (AMPGO) [55]; Basin Hopping (BH) [65]; Covariance Matrix Adaptation Evolution Algorithm (CMA-ES)[66]; Controlled Random Search with Local Mutation (CRS2) [67]; Differential Evolution (DE) [63]; Diving Rectangles Procedures (DIRECT) [68]; Firefly Algorithm [69]; Genetic Algorithm (GA) [70]; Multi-Level Single Linkage ALgorithm (MLSL) [71]; Particle Swarm Optimization (PSO) [72]; Shuffle Complex Evolution (SCE) [73]; and Simmulated Annealing (SA) [74].

Each algorithm was executed 100 times with distinct seeds and starting points. The stopping criteria was set to 2000 function evaluations or if the difference between the best solution found so far and the known global optimum is less than 10^{-6}. Table 3.15 shows the percentage of successful executions for algorithm in the 15 instances.

Table 3.15: Percentage of Successful execution of each algorithm.

Name	ABC+A-DVM	ABCX-m1+A-DVM	GBESTABC+A-DVM	MABC+A-DVM	AMPGO	ASA	BH	CMA-ES	CRS2	DE	DIRECT	FireFly	GA	MLSL	PSO	SCE	SA
Bukin06	0	0	0	0	94	0	0	0	0	0	0	0	0	0	0	0	0
Cola	0	0	0	0	58	0	0	0	0	0	0	0	0	0	0	0	0
CrossLegTable	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	9
CrownedCross	0	0	8	0	3	0	0	0	0	0	0	0	0	0	0	0	0
Damavandi	0	0	0	0	18	0	0	0	1	0	0	0	0	0	0	0	0
DeVilliersGlasser02	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Griewank	0	0	0	0	3	17	0	0	60	0	0	3	0	0	1	0	0
Rastrigin	0	0	0	0	97	73	81	6	88	0	0	49	0	0	31	48	0
Rosenbrock	0	0	0	0	100	0	0	100	67	0	100	51	0	100	0	100	0
Schewefel06	0	4	100	0	100	0	0	100	70	0	100	0	0	0	4	100	0
SineEnvelope	0	0	0	0	9	6	1	0	6	0	0	3	0	0	1	0	0
Trefethen	40	76	44	0	1	4	8	5	9	0	0	1	0	0	4	0	0
Whitley	0	0	0	0	5	2	28	3	24	0	0	0	0	0	2	0	0
XinSheYang03	0	0	3	0	5	3	0	0	0	0	0	5	0	0	0	0	0
Zimmerman	0	34	51	0	1	5	0	37	14	0	0	0	0	0	0	47	0

From the overall point of view, it can be understood that ABC based algorithms are slow to converge to the optimum. They failed to reach the optimum in instances where most of the others had no trouble, such as the Rastrigin and Rosenbrock instances. A possible reason would be that the ABC converges slowly in function whose landscape resembles a valley with ridges in the region close to the global optima.

On the other hand, the two ABC variants which biases the convergence the most (which could be seen as a good feature or not), was more successful than all other algorithms in four cases, CrossLegTable, CrownedCross, Trefethen and Zimmerman. Where, the first two are the third and fourth, respectively, most difficult functions in the entire benchmark suite. A possible hypothesis for the success in these cases could be due to the fact that both variants rely heavily on information from the overall best solution of the solution set in their position update steps, so the A-DVM guaranteed that every component of every solution would move towards the that solution in particular, speeding up the convergence of the solutions towards a local optima even further. Naturally, another explanation could might as well be attributed to the 'no-free lunch' [64] nature of the algorithms by themselves.

3.4 Concluding Remarks

In this chapter, a decision variable selection scheme named Adaptive Decision Variable Matrix (A-DVM) was proposed to be incorporated in the Artificial Bee Colony (ABC) algorithm. A-DVM was incorporated in both employed and or onlooker bees phases and can be used with any variant of the ABC, as well as other derivative-free algorithms that uses a solution set in its optimization process. A-DVM attempts to balance exploration and exploitation throughout the execution of the algorithm by constructing an augmented binary matrix that represents the choice of components of solutions in the solution set. The binary matrix is composed of a deterministic selection binary matrix that chooses matrix diagonals according to the proposal in [47] and another binary matrix whose components were selected by a random uniform distribution. The number of columns to be used from the deterministic matrix is determined by a self-adaptive parameter that is based on the Δ value, a measure of the sparsity of the actual solution set in the search space. Moreover, we also introduce a vector that stores the chosen solutions of $\boldsymbol{P_d}$ guarantees that every solution is a part of $\boldsymbol{P_a m}$ in the upcoming update step at least once before termination.

Influence of the A-DVM to the performance of the ABC when solving different families of multimodal unconstrained problems is verified by several numerical experiments where the A-DVM was incorporated to the original ABC as well as several variants. For all experiments, the 15 hardest instances from the Global Optimization Benchmark Suite [55] or a subset of those where chosen to be used.

The first numerical experiment had the objective to verify whether the A-DVM brought any improvement to ABC-based algorithms. A large budget of 10^5 function evaluation was assigned to each algorithm to be executed 30 times with distinct initial points. The A-DVM based algorithms were compared against their non A-DVM counterparts, and some representative heuristics such as the Particle Swarm Optimization (PSO), Evolutionary Particle Swarm Optimization (EPSO) and Differential Evolution (DE) to provide a baseline for the results. The results indicate that the A-DVM enhances the ability of the ABC to adapt to highly multimodal functions. However, the elimination of the full global search of the stochastic selection resulted in solutions not converging towards accumulation points that are located in basins, as seen in some instances where the A-DVM performed poorly.

The second numerical experiment consisted in testing the A-DVM in larger scale instances. The functions used in this experiment was a subset of 6 scalable instances of the 15 functions where their dimension was set to 100. We restrict the comparison to A-DVM and ABC algorithms only. A budget of 10^4 function evaluation was assigned to each algorithm to be executed 30 times with distinct initial points in the same fashion as the first experiment. The results suggested a small improvement in a restrict number of the 6 functions, pointing to the need of the A-DVM to be scalable as well, adapting itself to choose multiple decision variables instead of only one at a time.

Lastly, the third experiment aimed at testing A-DVM algorithms against a larger range of Derivative-Free Algorithms. The experiment used all 15 functions but with a more stringent budget of 2000 function evaluations and executed 100 times with distinct starting points. The results shown that although the A-DVM algorithms have slower convergence compared to some of the tailor-made algorithms, two versions that exploits the global best solution had much more successful execution than the others in four cases. This can be either attributed to the fact that the A-DVM biased the search even further, or simply to the no-free lunch theorem [64].

Chapter 4

A Nelder-Mead Direct Search with Simplex Gradients for Multimodal Optimization Problems

4.1 Introduction

Simplex gradients play a very important role in model based derivative-free methods, which approximates the objective function with a model function and then utilizes it to command the optimization process [75]. Simplex gradients, vastly used in the aforementioned methods, determine a descent direction of the actual objective function and can be defined even when f is non-smooth. Assuming any real-valued function objective function f in the same form of (2.1) where $f : \mathbb{R}^n \to \mathbb{R}$, let solution set $\boldsymbol{Y} = [\boldsymbol{y}^1,\ \boldsymbol{y}^2, \ldots,\ \boldsymbol{y}^k]$ where $\boldsymbol{y}^i \in \boldsymbol{Y}$ is ordered in ascending order based on $f(\boldsymbol{y}^i)$, and the matrix L of ℓ_1 length between solutions be defined as $L = L(\mathbf{Y}) := \begin{bmatrix}\boldsymbol{y}^1 - \boldsymbol{y}^0 \ \boldsymbol{y}^2 - \boldsymbol{y}^0 \ \cdots \ \boldsymbol{y}^k - \boldsymbol{y}^0\end{bmatrix} \in \mathbb{R}^{n\times k}$ for k arbitrary number of solutions in $\boldsymbol{Y}$. Moreover, let matrix δ_f (or column vector) of the difference of objective function values be defined as,

$$\delta_f = \delta_f(\mathbf{Y}) := \begin{bmatrix} (f(\boldsymbol{y}^1) - f(\boldsymbol{y}^0)) \\ \vdots \\ (f(\boldsymbol{y}^k) - f(\boldsymbol{y}^0)) \end{bmatrix}. \tag{4.1}$$

Where we assume that the columns $\boldsymbol{Y}$ are ordered. If L is a $n \times n$ invertible matrix, the simplex gradient of f with respect to $\boldsymbol{Y}$ falls into the determined case and is defined as,

$$\nabla_s f(\boldsymbol{Y}) = (L^\top)^{-1}\delta_f. \tag{4.2}$$

If the number of columns of L is greater than the rows, i.e., $k > n$ and $k+1$ columns are affinely independent, then the simplex gradient of f with respect to $\boldsymbol{Y}$ falls into the overdetermined case and it is calculated as the least squares solution of g,

$$\nabla_s f(\boldsymbol{Y}) = \operatorname{argmin}\left\{\left\|L^\top g - \delta_f\right\|^2\right\}. \tag{4.3}$$

Custodio and Vicente [76] showed that the simplex gradient is a good enough approximation of the real gradient if and only if the solution set in question is well poised. A set $\boldsymbol{Y}$ is well-poised if L has full rank. Since well-poisedness of a solution set is not easy to be guaranteed, the same authors introduced the concept of Λ-poisedness, an approximation of the well-poisedness. A set $\boldsymbol{Y}$ is Λ-poised if for an arbitrary value of Λ, $\boldsymbol{Y}$ can be decomposed in the following way,

$$\frac{\boldsymbol{Y}^\top}{\Delta} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top, \qquad \|\boldsymbol{\Sigma}\|_2 \leq \Lambda. \tag{4.4}$$

where $\Delta = \max_{1\leq i \leq n} \left\|\boldsymbol{y}^i - \boldsymbol{y}^0\right\|$ is the radius of the closed ball centered at $\boldsymbol{y}^0$ and $\|\cdot\|_2$ is the Spectral norm.

Although it has been shown that the Nelder-Mead performs well in multimodal optimization problems [77, 19, 31], its performance is directly tied to the initial point where it was initialized, or restarted. In an attempt to improve the robustness of the algorithm by trying to force it to converge to local optima regardless of the choice of starting point, we incorporate a more deterministic procedure, the estimation and calculation of simplex gradients, to the Nelder-Mead. We propose an adaptation of the simplex gradient to be integrated to the Nelder-Mead algorithm in the form of a polling step that is executed if the NM satisfies any termination test or if all geometric transformation steps fail to find a better point.

The inclusion of simplex gradients to a direct search method such as the Nelder-Mead has never been done before. We now denominate the modified Nelder-Mead algorithm, as the gradient-based Nelder-Mead (g-NM). A numerical experiment is performed to confirm our hypothesis, using the Moré, Garbow and Hillstrom [78], a suite of representative unconstrained optimization benchmark functions which features a vast class of optimization problems. The g-NM is compared against the classical Nelder-Mead, as well as against several representative population-based heuristics algorithms.

The chapter is structured in the following way, Section 4.2 details the finer steps of the g-NM. Section 4.3 explains the numerical experiments to assess the performance of the g-nm, while Section 4.4 discusses the results. Lastly Section 4.5 delineates some conclusions from the results obtained in the experiment.

4.2 A polling step to the Nelder-Mead

The polling process consists of constructing a set of points P_k, named polling set, in which f is evaluated for each point. A typical way of constructing P_k is as follows,

$$P_k = \{\boldsymbol{x}_k + \alpha_k \boldsymbol{b} \;:\; \boldsymbol{b} \in \boldsymbol{B}\} \tag{4.5}$$

where $\boldsymbol{x}_k$ is the current iterate (or incumbent solution), α_k the step size for the current iteration and $\boldsymbol{b}$ a direction vector from a positive spanning basis. Point in P_k are sampled following a specified order in order to find a point that decreases the objective function value in respect to the current iterate $f(\boldsymbol{x}_k)$. Understanding of positive spanning bases are paramount to the explanation of the polling procedure in the Nelder-mead, therefore we provide a brief foundation in the notion of positive span, positive spanning set and positive basis as follows.

Definition 4.2.1. Positive span [2, 79].
The positive span of a set of vectors $(v_1, \dots, v_n)^\top \in \mathbb{R}^n$ is the convex cone:

$$\text{pspan}(S) = \{\boldsymbol{v} \in \mathbb{R}^n \ : \ \boldsymbol{v} = \alpha_1 \boldsymbol{v}_1 + \dots + \alpha_r \boldsymbol{v}_r, \quad \alpha_i \geq 0, \ i = 1, \dots, r\}. \tag{4.6}$$

Definition 4.2.2. Positive spanning set.
A positive spanning set in $\mathbb{R}^n$ is a set of vectors whose positive span is $\mathbb{R}^n$. A set S is positively dependent if one a vector $\boldsymbol{v}_i \in S$ is in the convex cone pspan(S) spanned by the remaining vectors, i.e., $\boldsymbol{v}_i$ is a positive combination of the others; otherwise the set is positively independent.

Consequently, we have the following theorem:

Theorem 4.2.1. If set S spans $\mathbb{R}^n$ positively, then a subset $S - \boldsymbol{v}_i$ spans $\mathbb{R}^n$ linearly

Proof. Proofs of this theorem can be found in [2, 79, 80, 34]. □

Definition 4.2.3. Positive basis.
A positive basis $\boldsymbol{B}$ in $\mathbb{R}^n$ is a positively independent set whose positive span is $\mathbb{R}^n$. The number of vectors in $\boldsymbol{B}$ can range from any number between $n + 1$ and $2n$ [81]. A basis is called minimal if it has $n + 1$ vectors. On the other hand, it is called maximal if it is composed of $2n$ vectors.

Custodio and Vicente [76] showed that if the initial solution set is well-poised, ensuring that $\nabla_s f(\boldsymbol{Y})$ is a good enough approximation, then subsequent solution sets are also well-poised. This way, we must guarantee that in the Nelder-Mead the well-poisedness of the initial sampling set $\boldsymbol{Y}_0$, and by extension, the well-poisedness of $\boldsymbol{Y}_k$ at any iteration k by including a record $\boldsymbol{R}$ of all previously sampled points since $\boldsymbol{Y}_0$. Since heuristics to initialize $\boldsymbol{Y}_0$ in the context of the Nelder-Mead are diverse, and most of them provide no basis as to why it works, we suggest the construction of the initial solution set $\boldsymbol{Y}_0$ by matrix $\boldsymbol{W}$ as follows,

$$\boldsymbol{W} = \boldsymbol{Z}\boldsymbol{B}^-, \qquad \boldsymbol{B}^- = \left[\boldsymbol{e}_1 \, \boldsymbol{e}_2 \, \cdots \, \boldsymbol{e}_n \ - \sum_{i=1}^{n} \boldsymbol{e}_i \right]. \tag{4.7}$$

where $\boldsymbol{B}^-$ is a $n \times n + 1$ minimal positive spanning set and $\boldsymbol{Z}^{n \times n}$ is a diagonal matrix where the main diagonal is $\boldsymbol{z}^0 = U(\boldsymbol{l}, \boldsymbol{u})$, i.e., a point uniformly sampled within the lower and upper bound, l and u, respectively, of the search space of f. Since $\boldsymbol{W}$ is still a positive spanning basis, it has full row rank and presents another very useful property of positive spanning bases that has been shown by Regis [79], $\boldsymbol{W}$ is guaranteed to have a column $\boldsymbol{w}^i$ such that $\boldsymbol{w}^\top \boldsymbol{g} < 0$ where $\boldsymbol{g}$ is $\nabla_s f(\boldsymbol{Y})$, so $\boldsymbol{w}^i$ is a descent direction. Equipped with an initial solution set that is well-poised, we define the record of previous solutions $\boldsymbol{R}$ as a $n \times (n + 1) + k$ matrix as follows,

$$\boldsymbol{R} := [\boldsymbol{W} | \boldsymbol{Y}_k^\star] \qquad \boldsymbol{Y}_k^\star = \left[\boldsymbol{y}_1', \, \boldsymbol{y}_2', \, \dots, \boldsymbol{y}_k' \right]. \tag{4.8}$$

Where, $\boldsymbol{Y}_k^\star$ is a matrix that stores the point that was accepted by means of any of the geometric transformation steps (reflection, expansion or contraction) from the first until the k-th iteration. $\boldsymbol{R}$ is assumed to be ordered by objective function values in descending order so that (4.2) can be properly computed. The adapted polling step for the Nelder-Mead constructs a polling set P_k in the same fashion as in (4.5) with a few differences as follows,

$$P_k = \{\boldsymbol{y}_k + \alpha_k \boldsymbol{r}' \ : \ \boldsymbol{r}' \in \boldsymbol{R}\}. \tag{4.9}$$

Where $\boldsymbol{y}_k$ is $\boldsymbol{y}^0 \in \boldsymbol{Y}_k$ and descent direction $\boldsymbol{r}'$ is obtained from $\boldsymbol{R}$ by finding a direction $\boldsymbol{r}^i$ such that $(\boldsymbol{r}^i)^\top \boldsymbol{g} < 0$ where $\boldsymbol{g}$ is $\nabla_s f(\boldsymbol{Y})$ and normalizing it such that $\boldsymbol{r}' = \frac{\boldsymbol{r}^i}{\|\boldsymbol{r}^i\|}$. The line search that is used to build the polling set P_k sets step size α initial value as 10^{-4} and decreases it by a factor of $\tau = 0.5$ for each time $f(\boldsymbol{y}_k) < f(\boldsymbol{y}^p),\ \boldsymbol{y}^p \in P_k$. We define as a sufficient descent condition to halt the line search if a point $\boldsymbol{y}^\star$ is found where $f(\boldsymbol{y}^\star) < f(\boldsymbol{y}) - \epsilon$ for a small enough ϵ. For the sake of completion, we include Proposition 4.2.1 to endorse the well-poisedness of $\boldsymbol{R}$.

Proposition 4.2.1. Record set $\boldsymbol{R}$ is well-poised.

Proof. Since $\boldsymbol{R}$ is composed of $\boldsymbol{W}$ which is a minimal positive spanning basis with full row rank, $\boldsymbol{R}$ has full row rank, therefore it is well-poised. $\square$

After the construction of P_k, if no point whose objective function is lower than $\boldsymbol{y}_k$ is found, $\boldsymbol{y}_k$ is considered to be a local minimizer in the neighborhood around $\boldsymbol{y}_k$ with radius ϵ. If so, a new solution set $\boldsymbol{Y}_{k+1}$ should be initialized in any other point of the search space. Otherwise, if a $\boldsymbol{y}^\star$ where $f(\boldsymbol{y}^\star) < f(\boldsymbol{y}_k)$ is found, then $\boldsymbol{y}_k$ is not a local minimizer and a new solution set should be constructed centered at $\boldsymbol{y}^\star$. For the first case, we suggest the use of any heuristic to construct an initial solution set, from a new point uniformly sampled from the search space. In the second case, solution set $\boldsymbol{Y}_{k+1}$ is constructed by initializing a simplex $\boldsymbol{Y}_\star$ using $\boldsymbol{y}^\star$ as starting point using any starting simplex heuristic and performing a merge operation between $\boldsymbol{Y}_{k+1}$ and $\boldsymbol{Y}_\star$. Given two arbitrary $n \times n+1$ matrices $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ that represents a Nelder-Mead simplex, we define the merge operation from simplex $\boldsymbol{Y}_1$ resulting in a new simplex $\boldsymbol{Y}_n$ as,

$$\boldsymbol{Y}_n = \{\boldsymbol{y}_1^0, \boldsymbol{y}_1^1 + \gamma^m(\boldsymbol{y}_2^0 - \boldsymbol{y}_1^1), \boldsymbol{y}_1^2 + \gamma^m(\boldsymbol{y}_1^0 - \boldsymbol{y}_2^2), \boldsymbol{y}_1^n + \gamma^m(\boldsymbol{y}_2^{n-1} - \boldsymbol{y}_1^n)\}. \tag{4.10}$$

Where γ^m is a step length parameter empirically suggested to be 0.8. Additional evidence that shows some properties of the merge operation regarding the preservation of the volume of the simplex is given in Chapter 5.

In the new implementation of the Nelder-Mead, the polling step is used to verify two critical scenarios. The first, whether solution set $\boldsymbol{Y}_k$ has converged to a local optima $\boldsymbol{y}^\star$, i.e., $\|f(\boldsymbol{y}^i)\|_\infty \leq \epsilon,\ i = 1,\ \ldots,\ n+1$, such that $f(\boldsymbol{y}^\star) < f(\boldsymbol{y}_k^0)$ and $\boldsymbol{y}^\star \notin \boldsymbol{R}$ so that points in the closed ball $B = \{\boldsymbol{x} \in \mathbb{R}^n\ :\ \|\boldsymbol{x} - \boldsymbol{y}^\star\|_2 \leq \epsilon\}$ with $\boldsymbol{y}^\star$ as center, can be explored by the line search in the polling step. The second, is to confirm whether the solution set $\boldsymbol{Y}_k$ converged to a point $\boldsymbol{y}'$ such that $f(\boldsymbol{y}') > \operatorname{argmin}(\{f(\boldsymbol{y})\ :\ \boldsymbol{y} \in \boldsymbol{R}\}),\ \boldsymbol{y}' \notin \boldsymbol{R}$ or simply $\boldsymbol{y}' \in \boldsymbol{R}$. If so, that means that either the current solution set is stuck in a basin of attraction or that it converged to an already known accumulation point so that the $\boldsymbol{Y}_{k+1}$ may be restarted away from $\boldsymbol{y}'$ in a smart manner. Naturally, the polling step replaces the shrink operation and is executed if a condition of any termination test is satisfied indicating that the simplex has converged to a point or has collapsed to a subspace. The polling step in here differs from the former in the sense that, if a point $\boldsymbol{y}^\star$ is found then it replaces $\boldsymbol{y}^n$ and the current iteration is terminated with $\boldsymbol{Y}_k = \{\boldsymbol{y}^0,\ \boldsymbol{y}^1,\ \ldots,\ \boldsymbol{y}^\star\}$. The g-NM is described step-by-step in Algorithm 4.

4.3 Numerical Experiments

A numerical experiment is set to verify the performance and robustness of the g-NM for a range of multimodal functions. The main objective of this experiment is to answer the following research

Algorithm 4: Gradient-Based Nelder-Mead (g-NM) algorithm

```
Input: $f(\cdot)$, $\boldsymbol{x}_0$, $t$, $t'$, $\delta^{ic}$, $\delta^{oc}$, $\delta^{r}$, $\delta^{e}$, $\boldsymbol{z}_0$, $\alpha$
Output: $\boldsymbol{Y}_t = \{\boldsymbol{y}^0, \boldsymbol{y}^1, \cdots, \boldsymbol{y}^n\}$ where $\boldsymbol{y}^0 = \operatorname{argmin} f(\boldsymbol{y}),\ \boldsymbol{y} \in \boldsymbol{Y}_t$
Initialization: Initialize $\boldsymbol{Y}_0 = \boldsymbol{W}$ using $\boldsymbol{z}_0$, $\boldsymbol{R} \leftarrow \boldsymbol{W}$
1  for $i \leftarrow 1$ to $t$ do
2      if $\boldsymbol{Y}_i$ fails termination test then
3          $P_k \leftarrow$ PollingStep$(\boldsymbol{Y}_i, \alpha)$
4          $\boldsymbol{y}^\star \leftarrow \operatorname{argmin}(f(\boldsymbol{y})\ :\ \boldsymbol{y} \in P_k)$
5          if $f(\boldsymbol{y}^\star) < f(\boldsymbol{y}^0)$ and $\boldsymbol{y}^\star \notin \boldsymbol{R}$ then
6              $\boldsymbol{Y}_\star \leftarrow$ Initialization heuristic with $\boldsymbol{x}_0 = \boldsymbol{y}^\star$
7              $\boldsymbol{Y}_i \leftarrow$ MergeSimplex$(\boldsymbol{Y}_i, \boldsymbol{Y}_\star)$
8          else
9              Restart $\boldsymbol{Y}_i$ using a initialization heuristic with $\boldsymbol{x}_0 = \mathcal{U}(\boldsymbol{l}, \boldsymbol{u})$
10     else
11         $\boldsymbol{Y}_i \leftarrow \boldsymbol{Y}_{i-1}$
12     $\boldsymbol{Y}_i \leftarrow$ Order $(\boldsymbol{Y}_i)$
13     $\boldsymbol{c} \leftarrow$ ComputeCentroid $(\boldsymbol{Y}_i)$
14     $\boldsymbol{y}^{new} \leftarrow$ Reflection $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^r)$
15     $\boldsymbol{y}^{new} \leftarrow$ Expansion $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^e)$
16     $\boldsymbol{y}^{new} \leftarrow$ OutsideContraction $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^{oc})$
17     $\boldsymbol{y}^{new} \leftarrow$ InsideContraction $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^{ic})$
18     if no $\boldsymbol{y}^{new}$ is accepted then
19         $P_k \leftarrow$ PollingStep$(\boldsymbol{Y}_i, \alpha)$
20         $\boldsymbol{y}^\star \leftarrow \operatorname{argmin}(f(\boldsymbol{y})\ :\ \boldsymbol{y} \in P_k)$
21         if $f(\boldsymbol{y}^\star) < f(\boldsymbol{y}^0)$ and $\boldsymbol{y}^\star \notin \boldsymbol{R}$ then
22             $\boldsymbol{y}^{new} \leftarrow \boldsymbol{y}^\star$
23             $\boldsymbol{Y}_i \leftarrow (\boldsymbol{Y}_i \cup \boldsymbol{y}^\star) - \boldsymbol{y}^n$
24             $\boldsymbol{R} \leftarrow \boldsymbol{R} \cup \boldsymbol{y}^\star$
25         else
26             Restart $\boldsymbol{Y}_i$ using a initialization heuristic with $\boldsymbol{x}_0 = \mathcal{U}(\boldsymbol{l}, \boldsymbol{u})$
27             $\boldsymbol{R} \leftarrow \boldsymbol{R} \cup \operatorname{argmin}(\{f(\boldsymbol{y}) :\ \boldsymbol{y} \in \boldsymbol{Y}_i\})$
28     else
29         $\boldsymbol{R} \leftarrow \boldsymbol{R} \cup \boldsymbol{y}^{new}$
30 end
31 return $\boldsymbol{Y}_t$
```

question: ”Does using simplex gradients as a polling step procedure in the Nelder-Mead result in an improvement of the search capabilities of the algorithm, for multimodal instances of small dimensionality?”. As a testbed, the Moré, Garbow and Hillstrom [78] benchmark suite was chosen because it covers a large class of multimodal problems. We chose instances with no more than 30 dimensions due to the known fact that the performance of Nelder-Mead based algorithms greatly deteriorates in instances where $n > 100$.

We divide the 40 functions into three subgroups of families of problems in accordance to Moré, Garbow and Hillstrom [78]: nonlinear systems of inequations, problems that involve polynomial systems of inequalities; nonlinear least squares, problems that resemble the classical minimization of the difference of squares; and unconstrained minimization, nonlinear problems characteristic of unconstrained minimization problems. The details of each are described in Tables 4.1, 4.2 and 4.3, each column denotes the name, number of decision variables, number of system of inequations and the global optima, if known, respectively. We establish the boundary of all problems to fall in the range $[-100, 100]$.

Table 4.1: Systems of nonlinear equation instances of the Moré-Garbow-Hillstrom suite.

Name	**n**	**m**	**Opt**
Rosenbrock	2	2	0
Powell singular	16	16	0
Powell badly scaled	2	2	0
Wood	4	4	0
Helical valley	3	3	0
Chebyquad	9	5	0
Brown almost-linear	30	30	0
Discrete boundary	30	30	0
Discrete Integral	30	30	0
Trigonometric	10	10	0
Variably dimensioned	10	10	0
Broyden tridiagonal	10	10	0
Broyden banded	10	10	0

We test the performance of the g-nm against the original Nelder-Mead, as well as against a selection of representative Population-based heuristics. We limit our scope solely against population-based heuristics for two main reasons. The first is to do a direct comparison of a direct search method against a population heuristics. Second, because literature on model-based or direct search derivative-free techniques which tested their propositions using the entire benchmark is nonexistent, limited only to population based heuristics. The chosen algorithms are some of the most representative of the family, as well as some famous variants of them. Of the chosen heuristics, the canonical and variants of the ABC from Chapter 3: ABC, ABC-ES, ABCX-m1 and ABCX-m5 as well as their A-DVM counterparts were included in the experiment. Moreover, we include the Particle Swarm Optimization from Kennedy and Eberhart [61] and some of its variants, such as the Evolutionary

Table 4.2: Nonlinear Least Squares instances of the Moré-Garbow-Hillstrom suite.

Name	n	m	Opt
Linear full rank	5	4	$m-n$
Linear rank 1	5	4	$\frac{m(m-1)}{2(2m+1)}$
Linear rank 1 zero	10	9	$\frac{m^2+3m-6}{2(2m-3)}$
Freudstein and Roth	2	2	0, 48.9842
Bard	3	15	$8.21487\ldots10^{-3}$
Kowalik and Osborne	4	11	$3.07505\ldots10^{-4}$
Meyer	3	16	87.9458
Watson	9	31	$1.39976\ldots10^{-6}$
Box three-dimensional	3	5	0
Jennrich and Sampson	2	2	126.362
Brown and Dennis	4	4	85822.2
Osborne 1	5	33	$5.46489\ldots10^{-5}$
Osborne 2	11	65	$3.01377\ldots10^{-2}$
Quadratic	10	10	0

Table 4.3: Unconstrained optimization instances of the Moré-Garbow-Hillstrom suite.

Name	n	m	Opt
Biggs EXP6	6	6	0
Gaussian	3	15	$1.12793\ldots10^{-8}$
Penalty I	10	9	$7.08765\ldots10^{-5}$
Penalty II	10	20	$9.37629\ldots10^{-4}$
Brown badly scaled	2	3	0
Gulf R&D	3	20	0
Extended Rosenbrock	10	10	0
Extended Powell	16	16	0
Beale	2	3	0
Dixon	5	5	0
GaoHanAlmostQuadratic	2	2	0
McKinnon	2	1	-2.5
OrenPower	2	1	0

Particle Swarm Optimization by Miranda and Fonseca [62], Maximum Search Limitation Evolutionary Particle Swarm Optimization by Neto et al. [82] (MS-EPSO) and the Quantum behaved Particle Swarm Optimization (QPSO) by Sun et al. [83]. The Differential Evolution (DE) [63], Genetic Algorithm from Holland [70] (GA) and Covariance-Matrix Evolutionary Strategy (CMA-ES) from Hansem et. al [66] are also part of the numerical experiment.

The experiment was conducted in a machine with the following hardware configuration: Intel core i7-6700 ”Skylake” 3.4 GHz CPU; 16 GB RAM DDR4 3200 clocked at 3000 MHz. The running operating system (OS) is UbuntuOS 18.04. All algorithms were written in the python 3 programming language. Floating point operations were handled by the numpy package, version 1.19.1.

Each algorithm is executed 30 times. The stopping criteria was set to 10^5 function evaluations (FE's) or if the difference between the best value found so far and the global optimum $f(\boldsymbol{x}^*)$ is less than 10^{-6}. The solution set size was fixed for all population based heuristics at 50. For PSO and QPSO, the inertia factor (w_1) was set to 0.6 and both cognitive and social parameters (w_2, w_3) to 1.8. For EPSO, the number of replicas is set to 1, communication and mutation ratio to 0.8. For Differential Evolution (DE) [63] with *best1bin* strategy, *F* value was 0.5 and *CR* 0.9. The Genetic Algorithm uses two-point crossover with crossover rate of 0.8 and mutation rate of 0.2 while keeping 10% of the best solutions at each generation. The CMA-ES uses the $(\mu + \lambda)$ strategy and uses a Rank-one update to its covariance matrix. Every ABC uses the same set of parameters from Chapter 3. Both NM and g-NM use the set of adaptive parameters from Gao and Liu [26] where $\delta^r = 1$, $\delta^e = 1 + \frac{2}{d}$, $\delta^{oc} = 0.75 - \frac{1}{2d}$ and $\gamma^s = 1 - \frac{1}{d}$.

4.4 Computational Results

Due to the sheer size of the experiment, we summarize the results in the following way, Table 4.4 shows the top 5 algorithms which had the lowest mean in descending order for each instance. Empty entries represent instances where the majority of the algorithms (more than 80%) achieved the same mean for all 30 runs. We also summarized the statistical relevance of the data in Table 4.5 by running a pairwise Wilcoxon signed ranked test with significance set to 95% using the g-NM against all other algorithms. Entries with a '$+$' sign mean that statistical significance was found ($p \leq 0.05$) while entries with a '$-$' sign mean that no significance was found ($p > 0.05$). Figure 4.1 show the percentage of the best mean of the runs for each algorithm and the rank of each algorithm for each class of problem.

Firstly, it is possible to observe that in 16 instances, all algorithms achieved the same mean, i.e., that they were able to reach the optimum for all 30 executions. From those 16 problems, 3 are from the family of nonlinear system of equations, 5 are from the nonlinear least-squares and 7 are from the unconstrained optimization family. We conclude that this subgroup of problems are ”entryways”, that is, a mean to verify whether a derivative-free algorithm can at least solve these problems and have the same performance of the rest.

Secondly, we point out that for four problems in particular: Brown Almost linear; Jennrich and Sampson; Kowalik and Osborne; and Osborne I the g-NM achieved the lowest mean together with the majority of the algorithms. On the other hand the g-NM mean value was lower in relation to the original Nelder-Mead, which is corroborated by the statistical significance from the differences

Table 4.4: Top five algorithm ranking by mean for each problem.

Function	First	Second	Third	Fourth	Fifth
Bard	-	-	-	-	-
Beale	-	-	-	-	-
BiggsEXP6	CMA-ES	ABCX-m1	g-NM	EPSO	QPSO
Box3D	-	-	-	-	-
BrownAlmostLinear	g-NM*	CMA-ES*	MS-EPSO*	DE*	GA
BrownAndDennis	g-NM	CMA-ES	ABCX-m5	ABCX-m1	ABC
BrownBadlyScaled	ABCX-m1+A-DVM	ABCX-m5+A-DVM	EPSO	DE	GA
BroydenBanded	-	-	-	-	-
BroydenTridiagonal	ABCX-m1	g-NM	DE	GA	CMA-ES
Chebyquad	EPSO	QPSO	ABCX-m1+A-DVM	ABCX-m1	MS-EPSO
DiscreteBoundary	-	-	-	-	-
DiscreteIntegral	-	-	-	-	-
Dixon	-	-	-	-	-
ExtendedPowellSingular	g-NM	NM	GA	CMA-ES	ABCX-m5
ExtendedRosenbrock	g-NM	NM	CMA-ES	ABCX-m5	ABCX-m1
FreudsteinAndRoth	-	-	-	-	-
GaoHanAlmostQuadratic	-	-	-	-	-
Gaussian	-	-	-	-	-
GulfR&D	-	-	-	-	-
HelicalValley	g-NM	NM	CMA-ES	GA	QPSO
JennrichAndSampson	g-NM*	PSO*	EPSO*	DE*	MS-EPSO*
KowalikAndOsborne	g-NM*	DE*	CMA-ES*	EPSO*	QPSO*
LinearFullRank	g-NM	NM	ABCX-m5	GA	EPSO
LinearRank1	-	-	-	-	-
LinearRank1ZeroColumnsAndRows	-	-	-	-	-
McKinnon	g-NM*	NM*	CMA-ES*	QPSO	PSO
Meyer	MS-EPSO	EPSO	QPSO	ABCX-m1	ABC
OrenPower	-	-	-	-	-
Osborne1	g-NM*	EPSO*	PSO*	ABCX-m1*	ABCX-m5*
Osborne2	ABC+ES	ABCX-m1+A-DVM	ABCX-m1	DE	g-NM
Penalty1	g-NM*	NM*	MS-EPSO*	QPSO*	ABC*
Penalty2	-	-	-	-	-
PowellBadlyScaled	-	-	-	-	-
PowellSingular	CMA-ES	ABCX-m1	g-NM	NM	PSO
Quadratic	g-NM	CMA-ES	NM	ABCX-m5	ABCX-m1
Rosenbrock	g-NM	NM	CMA-ES	ABCX-m1	ABCX-m5
Trigonometric	EPSO*	g-NM*	MS-EPSO*	PSO*	CMA-ES*
VariablyDimensioned	EPSO*	CMA-ES*	g-NM*	GA*	MS-EPSO*
Watson	MS-EPSO	g-NM	ABCX-m1	EPSO	DE
Wood	g-NM	ABCX-m5+A-DVM	DE	ABCX-m1+A-DVM	MS-EPSO

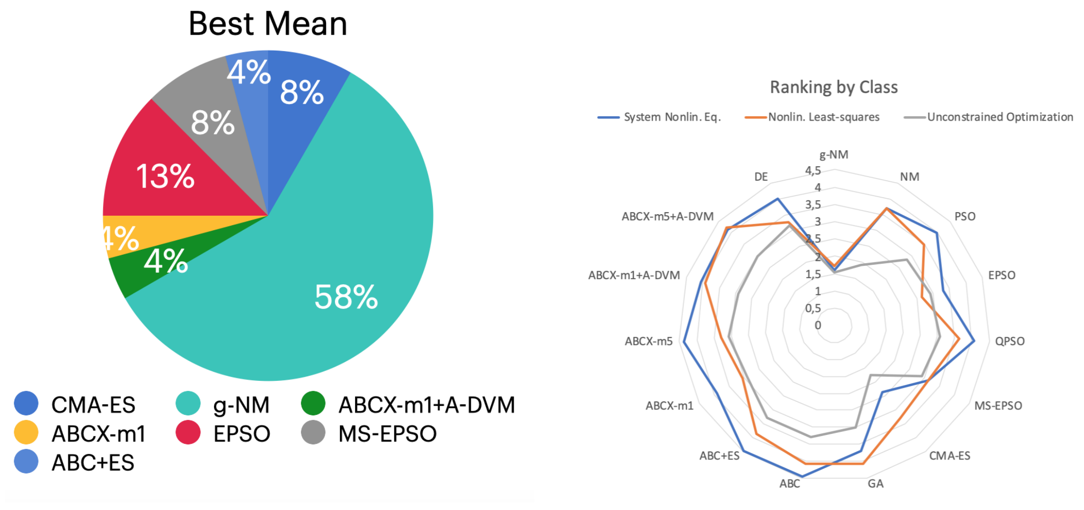


Figure 4.1: Percentage of the best overall mean and radar plot of the ranking of each technique according to the class of problem.

Table 4.5: Pairwise Wilcoxon sign-rank test of the g-NM against all other algorithms.

Function	NM	ABC	ABC-ES	ABCX-m1	ABCX-m5	ABC+A-DVM	ABCX-m1+A-DVM	ABCX-m5+A-DVM	PSO	EPSO	MS-EPSO	QPSO	DE	GA	CMA-ES
F1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F3	+	+	+	+	+	+	-	-	+	-	-	+	+	-	+
F4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F5	+	+	+	+	-	+	+	+	+	+	-	+	-	+	-
F6	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F7	-	+	+	-	-	-	+	+	-	+	+	+	+	+	+
F8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F9	+	+	+	-	+	+	+	+	+	+	+	+	+	-	+
F10	-	-	-	+	+	+	+	+	-	+	+	+	-	-	+
F11	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F14	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F15	-	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F20	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F21	+	+	+	+	+	+	+	+	-	-	-	-	-	+	+
F22	+	+	+	+	+	+	+	+	+	-	-	-	-	-	-
F23	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F24	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F25	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F26	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F27	-	-	-	+	+	+	+	+	+	+	+	+	-	-	+
F28	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F29	+	+	+	-	-	+	+	+	-	-	-	+	+	-	-
F30	+	-	+	+	+	+	+	+	+	+	+	+	-	-	-
F31	-	-	+	+	-	-	+	-	+	-	-	-	-	-	-
F32	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F33	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
F34	-	+	+	-	+	+	+	+	-	+	+	-	+	+	-
F35	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F36	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
F37	+	+	+	+	+	+	+	+	-	-	-	-	+	+	-
F38	+	+	+	+	+	+	+	+	+	+	-	-	-	+	-
F39	+	+	+	-	+	+	-	+	-	-	-	+	-	+	+
F40	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+

of the means ($p < 0.05$). Out of the five instances, three are from the Nonlinear least-squares, and one from the nonlinear system of equation families. From this observation, it is possible to assert that the inclusion of polling improved the robustness of the NM, at least in problems from the multimodal nonlinear-least squares minimization instances.

We now change our focus to the 10 cases where the g-NM was more robust in relation to all other algorithms backed up by statistical evidence: Brown and Dennis, Extended Powell Singular; Extended Rosenbrock; Helical Valley; Linear full rank; McKinnon; Quadratic; Rosenbrock; and Wood. Out of the nine instances, three belong to the first, second and third classes of problems. There are two possible causes that explain the reason why the g-NM was more robust than the other algorithms. The first, is that since most of the problems where the g-NM was better featured low dimensionality ($n < 5$), the NM itself can properly adapt the simplex to the contour lines of the objective function landscape, as pointed by Lagarias [19]. Therefore the simplex gradient itself contributed very little to the robustness of the g-NM. This behavior can be seen in the Brown and Dennis, Extended Powell, Extended Rosenbrock, Helical Valley, Quadratic and Rosenbrock. The same can be said for the well-known McKinnon instance, a strictly convex function. The second cause is that indeed finding a descent direction has enabled the solutions of the g-NM to displace themselves farther from deceptive local optima, as seen in the Wood and Linear full rank instances.

The last 10 cases are separated into two categories, where the g-NM was in the top five but not necessarily in the first place, and where the g-NM was not in the top five. The first category is comprised of 7 instances: Biggs exponential; Broyden tridiagonal; Osborne 2; Powell singular; Trigonometric; Variably dimensional and Watson. Four from the first class, two from the second and one from the third. The second class consists of three functions: Brown Badly scaled, Chebyquad and Meyer. Two from the first class and one from the second class. We limit our explanation to the second group because any explanation regarding the first group can be reduced to the "no free lunch" theorem. The most plausible reason to justify the performance of the g-NM regarding the second group is that the three problems consist of solving a system of polynomial inequations on a nonconvex search space where the objective function landscape has ridges around its critical region, which could have possibly resulted in the g-NM computing descent directions towards deceptive local optima.

4.5 Concluding Remarks

This chapter introduced a new version of the Nelder-Mead direct search method that computes an approximation of simplex gradients to to be used in the polling step in order to thoroughly explore the region around local accumulation points. The polling step replaces the shrink step and is also called if the solution set is flagged to be restarted. In the polling, a polling set P_k is constructed by a a line search using as descent direction, a unit vector from record set $\boldsymbol{R}$ that is found by the calculation of simplex gradients. To guarantee that the simplex gradient is a good enoguh approximation of the true gradient, well poisedness of the sampling set $\boldsymbol{R}$ is guaranteed by initializing the first solution set $\boldsymbol{Y}_0$ with a Positive spanning basis $\boldsymbol{W}$. After points of P_k are evaluated, if a point $\boldsymbol{y}^\star$ is found that is better than the current best iterate, the current solution set $\boldsymbol{Y}_k$ is shifted by creating another solution set $\boldsymbol{Y}_\star$ centered at $\boldsymbol{y}^\star$ and performing a merging operation between the two set. If no point is found the current best is flagged as a local attractor and a solution set $\boldsymbol{Y}_{k+1}$ is constructed from another point sampled from a uniform distribution which is not part of $\boldsymbol{R}$.

A numerical experiment was performed to answer the research question posed earlier in this chapter, if including polling procedures by computing simplex gradients to Nelder-Mead algorithm would result in a new algorithm that is more robust than the original in at least a subgroup of multimodal functions. The experiment consisted of testing the g-NM against the original Nelder-Mead as well as several representative population based heuristics using the Moré-Garbow-Hillstron [78] suite of 40 benchmark functions. These functions were separated into three subgroups in accordance to the same authors: systems of linear inequations; nonlinear least squares; and unconstrained nonlinear optimization. Results and statistical evidence indicated that the novel approach was more reliable than the base algorithm in all instances, achieved the best results in 10 instances, and had the worst performance in the system of linear inequation class of problems. Therefore, it can be said that the g-NM is indeed robust for small scale nonlinear least squares and unconstrained optimization problems in comparison to some well-known heuristics. However, this assertion cannot be made to larger problems ($n > 100$) since it was not in the scope of this experiments. Consequently, further adaptation and testing in such problems is a promising direction for future works.

Chapter 5

Deterministic Derivative-Free algorithms for Constrained Optimization Problems

5.1 Introduction

Problems featuring equality and inequality constraints are mathematical models which better approximates real life scenarios. Naturally, the more constraints a problem has, the more stringent is the feasible region, whether linear or nonlinear. Assuming f is nonlinear and the search space is $\mathcal{C}^0$, i.e., nondifferentiable and non Lipschitz continuous, the standard form of a nonlinear optimization problem with constraints is,

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}) \leq 0, \qquad i = 1, \ldots, p \\ & h_i(\boldsymbol{x}) = 0, \qquad i = 1, \ldots, m \\ & \boldsymbol{l}_j \leq \boldsymbol{x}_j \leq \boldsymbol{u}_j, \quad j = 1, \ldots, n, \end{aligned} \tag{5.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^p$, $h : \mathbb{R}^n \to \mathbb{R}^m$. A typical example of (5.1) are engineering design optimization problems. Engineering design is a challenging class of problem that consists of optimizing the shape of the design of a device/part while keeping its standards. These problems are modeled as unconstrained nonlinear optimization problems. Moreover, the constraints that appear in these problems are typically nonlinear [1].

Because of the nature of those problems, algorithms that rely on a model constructed from the first or second derivative cannot be used at any instance, leaving way for derivative-free algorithms to tackle these problems firsthand. However, the majority of derivative-free algorithms were intended for unconstrained optimization, so modifications are required so that they can handle problems from that family.

In this chapter, we propose several modifications to two derivative-free algorithms, the Nelder-Mead direct search, seen in Chapter 4 and the A-DVM based Artificial Bee Colony (ABC) algorithm from Chapter **??**. The modifications not only allow these algorithms to solve constrained nonlinear problems, but they are also an attempt to introduce deterministic procedures to two derivative-free algorithms that rely on randomness so that they can be invariant of starting point or random seeds. Our contributions are threefold: (1) Development of an augmented Lagrangian penalty method for algorithms that feature solution sets along with a penalty barrier method to be able to handle equality and inequality constraints where the Lagrange multipliers as well as the barrier multipliers are updated every iteration; (2) A new version of the Nelder-Mead to solve constrained problems (c-NM) that guarantees that a nondegenerate simplex is built in the early stages of the algorithm, and if it ever degenerates, it is equipped with safeguards so that the nondegeneracy property is maintained; (3) An extension of the A-DVM technique to be used with the ABC algorithm that employs additional to choose decision variables according to the degree of satisfiability.

A numerical experiment is carried out to measure the performance of the proposed algorithms. Eight constrained functions from the field of Engineering Design are chosen as a base of comparison. The algorithms are tested against several representative derivative-free algorithms and their variants and are compared to the most prominent results found in the literature. results suggests a favorable outcome to both algorithms in the majority of the cases.

This chapter structure is divided as follows. Section 5.2 explains the adapted augmented lagrangian barrier method for derivative-free algorithms. Section 5.3 discusses the changed to the A-DVM based Artificial Bee colony for constrained problems, while Section 5.4 details the modified Nelder-Mead algorithm to handle constrained problems. Details and configurations of the numerical experiment, as well as the formulation of each optimization problem is shown in Section 5.5. Section 5.6 discusses the results and statistical analysis. Lastly, Section 5.7 draws conclusions and highlights future directions.

5.2 Adapting the Augmented Lagrangian for Solution Sets of Derivative-Free Algorithms

Most Derivative-Free algorithms are ill-equipped to handle problems of the form of (5.1) as they are and even more for nonlinear constraints are nonlinear due to the fact that they do not incorporate any information about feasible regions to the solution set. One way to correct this is including a penalty or barrier function to penalize infeasible solutions [41]. Penalty methods to derivative-free algorithms is a very developed topic thoroughly studied by researchers. Literature on penalty methods is very rich, the reader is encouraged to read the survey of Mezura-Montes [84] which summarizes constraint methods for evolutionary computation algorithms that are also commonplace to other families of derivative-free algorithms.

One of the simplest approach is to enforce the decision variables to stay within their feasible bounds using a forcing function ϕ,

$$\phi(y_i) = \begin{cases} u_i & \text{if } y_i > u_i \\ l_i & \text{if } y_i < l_i \end{cases} \tag{5.2}$$

although incorporating a barrier method such as (5.2) to any Derivative-Free method to solve problems in the form of (5.1) is a simple and straightforward way to remedy the situation, it would bring several problems to the performance of any algorithm. Consequences of using (5.2) to population heuristics such as the Artificial Bee Colony and a direct search method such as the Nelder-Mead are discussed in Section 5.4 and Section 5.3, respectively.

Rather than (5.2), a better alternative is to use methods that allow solutions to be updated with a relative freedom but that penalize the objective function value of infeasible solutions. We highlight the usage of one such method, the augmented lagrangian with dislocated penalties [41, 85] to reformulate (5.1) to an unconstrained problem which penalizes infeasible solutions according to weights that are updated iteratively. Equality and non-equality constraints are moved to the objective function to be penalized by multipliers in the following formulation at any iteration k as follows,

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) + \frac{\rho^k}{2}\left[\left\|h(\boldsymbol{x}) + \frac{\bar{\boldsymbol{\lambda}}^k}{\rho^k}\right\|^2 + \left\|g(\boldsymbol{x}) + \frac{\bar{\boldsymbol{\mu}}^k}{\rho^k}\right\|_+^2\right] \\ &\text{subject to} && \boldsymbol{l}_j \leq \boldsymbol{x}_j \leq \boldsymbol{u}_j, \quad j = 1, \dots, n. \end{aligned} \tag{5.3}$$

Where ρ^k is the Lagrange multiplier, $\bar{\boldsymbol{\lambda}}$ and $\bar{\boldsymbol{\mu}}$ are the equalities and inequalities multipliers, respectively. Assuming that the family of problems in the form of (5.1) are $\mathcal{C}^0$ an the constraints are nonlinear, calculation or approximations of ∇h or ∇g are not available, therefore it is not viable to check for constraint qualification conditions (e.g. KKT, Mangasarian-Fromovitz) [85]. The multipliers ρ^k, $\bar{\boldsymbol{\lambda}}$ and $\bar{\boldsymbol{\mu}}$ start from an initial value and are updated each iteration so that the penalties may be increased or decreased according to the feasibility of solutions. The update step is as follows, firstly, compute vector $\boldsymbol{V}^k$ of the inequalities dislocation,

$$V_i^k = \max\left\{g_i(x_k), -\frac{\bar{\boldsymbol{\mu}}_i^k}{\rho^k}\right\}, \quad i = 1, \dots, p. \tag{5.4}$$

If $\max\left\{\ \|h(x^k)\|_\infty, \|V^k\|_\infty\right\} \leq \tau \max\left\{\ \|h(x^{k-1})\|_\infty, \|V^{k-1}\|_\infty\right\}$, set $\rho_{k+1} = \rho^k$, otherwise set $\rho_{k+1} = \gamma\rho^k$. It is advisable to set rules for the increase of multiplier ρ^k, either establishing a limit ρ_{max}, or resetting ρ^k to a more acceptable value once it reaches a threshold value. We opted for the last in our formulation. After updating ρ^k, suitable values for $\bar{\boldsymbol{\lambda}}^k$ and $\bar{\boldsymbol{\mu}}^k$ are computed,

$$\begin{aligned} \bar{\lambda}_i^{k+1} &= \min\left\{\lambda_{max}, \max\{\lambda_{min}, \bar{\lambda}_i^k + \rho^k h_i(x^k)\}\right\}, \qquad i = 1, \dots, m \\ \bar{\mu}_i^{k+1} &= \min\left\{\mu_{max}, \max\{0, \bar{\mu}_i^k + \rho^k g_i(x^k)\}\ \right\}, \qquad i = 1, \dots, p \end{aligned} \tag{5.5}$$

where $\lambda_{min}, \lambda_{max}$ and μ_{min} are user defined parameters sensitive to each problem instance. Because the Lagrangian multipliers are more suited to methods that use only one incumbent solution at a time, multipliers ρ, μ, λ and β are reset to their starting values if the solution set had their values restarted. We extend the augmented lagrangian form of (5.3) to account for solutions that are out of their box constraints, so that it can properly penalize solutions that satisfies constraints $g(\boldsymbol{x})$ and $h(\boldsymbol{x})$ but are out of their bounds, resulting in the following formulation,

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad f(\boldsymbol{x}) + \frac{\rho^k}{2}\left[\left\|h(\boldsymbol{x}) + \frac{\bar{\boldsymbol{\lambda}}^k}{\rho^k}\right\|^2 + \left\|g(\boldsymbol{x}) + \frac{\bar{\boldsymbol{\mu}}^k}{\rho^k}\right\|_+^2\right] + \|z(\boldsymbol{x})\|^2. \tag{5.6}$$

Where penalty function $z(\boldsymbol{x})$ relies on a weight factor $\beta^k = [\beta_{min}, \beta_{max}]$ that acts as a multiplier like λ_k and μ_k going by the following rule,

$$z(\boldsymbol{x}) = \chi(\beta^k) + \beta^k \sum_{i=1}^{n} z(x_i) = \begin{cases} l_i + x_i & \text{if } x_i < l_i, \\ x_i - u_i & \text{if } x_i > u_i, \\ 0, & \text{otherwise,} \end{cases} \qquad \text{for } i = 1, \ldots, n.$$
$$\chi(\beta^k) = \begin{cases} \beta^k & \text{if } \sum_{i=1}^{n} z(x_i) > 0, \\ 0, & \text{otherwise,} \end{cases} \tag{5.7}$$

Multiplier β^k is updated similarly to ρ^k: the condition $\left\|z(x^k)\right\|_\infty \leq \left\|z(x^{k-1})\right\|_\infty$ is verified, if it is satisfied then $\beta^{k+1} = \beta_k$, otherwise $\beta^{k+1} = \gamma^b \beta_k$, $\gamma^b > 1$. This way, formulation (5.6) not only penalizes solutions that breaks constraint feasibility, but also solutions that are out of the boundary of the search space. We now describe further modifications to the A-DVM based Artificial Bee colony and the Nelder-Mead algorithm using (5.6) for constrained nonlinear problems with nonlinear constraints.

5.3 A-DVM based Artificial Bee Colony algorithm for constrained optimization

One of the most interesting point of population heuristics is that they can be employed for a multitude of problems without the need of a single modification. That said, the Artificial Bee Colony using the A-DVM could might as well be used as it is to solve problems in the likes of (5.1) with the simple addition of a barrier function like (5.2). We disprove this hypothesis with an assertion found in many books on global optimization when speaking about constrained optimization problems [41, 1, 44]: "Local optima are generally found from solutions that started from the infeasible region which iteratively moved to the feasible region". Furthermore, in [1] and [84], the authors state that local optima of problems of the form of (5.1) usually lie in the boundary of the feasible region.

In light of these affirmations, additional deterministic rules are added to the A-DVM to take full advantage of these properties together with changing the problem formulation from (5.1) to (5.6) to penalize infeasible solutions. The new rules are applied to the construction of deterministic matrix $\boldsymbol{P}_d$ and take precedence over the standard procedure of selecting diagonals. They take advantage of the formulation of some constraints so that feasible solutions can be found easily and are as follows,

1. **Rule 1:** If a constraint g_j or h_j depend on only one component x_j of $\boldsymbol{x}$. Furthermore, x_j violates g_j, then x_j is chosen to be updated. Otherwise, if g_j or h_j depend on multiple variables and is the only constraint that was not satisfied, a variable $x_{j'}$ associated to g_j is chosen.

2. **Rule 2**: In order to intensify the local search in the vicinity of the feasible bounds, a component x_j of $\boldsymbol{x}$ whose euclidean distance are less than a predetermined value ϵ to either u_j or l_j is chosen. ϵ is seen as the radius of an euclidean ball $B = \{\boldsymbol{y} \in \mathbb{R}^n : \|\boldsymbol{y} - \boldsymbol{x}\|_2 \leq \epsilon\}$, centered at $\boldsymbol{x}$.

Final calculation of P_{am} follows the same steps as in Algorithm 3. The step-by-step of the A-DVM for constrained problems is detailed in Algorithm 5.

Algorithm 5: Adapted A-DVM for constrained problems

```
1  $\varDelta \leftarrow$ ComputeDelta($\boldsymbol{X}$);
2  $\alpha \leftarrow \rho\left((1-\varDelta)K_1 + \varDelta K_2\right)$;
3  $\beta \leftarrow 1-\alpha$;
4  $\boldsymbol{P_r} \leftarrow$ BuildRandomMatrix($\beta$);
5  $\boldsymbol{P_d} \leftarrow$ BuildDeterministicMatrix($\alpha, \boldsymbol{H}$);
6  $\boldsymbol{P_d} \leftarrow$ FindSingleViolatedConstraint($g, h$);
7  $\boldsymbol{P_d} \leftarrow$ CheckBoundaryDistance($l, u, \epsilon$);
8  if $\boldsymbol{H} = (1,\dots,1)$ then
9      $\boldsymbol{H} \leftarrow (0,\dots,0)$;
10 $\boldsymbol{H} \leftarrow$ UpdateHistory($\boldsymbol{H}, \boldsymbol{P_d}, \boldsymbol{P_{am}}$)    ▷ Index of columns of $\boldsymbol{P_d}$ in $\boldsymbol{P_{am}}$ are 1 in $\boldsymbol{H}$;
11 $\boldsymbol{P_{am}} \leftarrow \beta\boldsymbol{P_r} \oplus \alpha\boldsymbol{P_d}$;
12 $\boldsymbol{c} \leftarrow$ ChooseVariables($\boldsymbol{X}, \boldsymbol{P_{am}}$);
13 $\boldsymbol{c}' \leftarrow$ UpdateStep($\boldsymbol{c}$)    ▷ Use any update function;
14 $\boldsymbol{X} \leftarrow$ UpdateSolutions($\boldsymbol{X}, \boldsymbol{c}'$);
15 $\boldsymbol{X} \leftarrow$ SafeguardStep($\boldsymbol{X}$);
```

5.4 A Nelder-Mead for constrained optimization

The original implementation of the Nelder-Mead algorithm is ill equipped to handle problems like (5.1). We now refer to the solution simplex as the $n+1$ simplex represented by the solution set $\boldsymbol{Y}_k$. Excluding the fact that it was made for unconstrained optimization, it also lacks several things: proper termination tests to verify whether the solution set has converged to a point that lies in infeasible regions; restart procedures that takes into account the feasibility of the points of the solutions set; fitting the simplex to the feasible region of the problem when at least one point of the solution set lies in the feasible region; preventing unnecessary shrink steps that degenerate the volume of the simplex.

We begin by arguing that using a hard barrier function like (5.2) when evaluating the solution set of the Nelder-Mead violates Definition 2.2.1 except in the unidimensional case. Proposition 5.4.1 provides a strong argument against methods that force solutions to stay in their bounds.

Proposition 5.4.1. Consider a problem in the form of (5.1) where $f:\mathbb{R}^n \to \mathbb{R},\ n>1$. Then any nonshrink iteration using barrier method (5.2) that ends up with any $y_i > u_i$ or $y_i < l_i,\ i=1,\dots,n$, changes the volume of the simplex to $|\kappa|\text{vol}(\boldsymbol{Y}) < |\tau|\text{vol}(\boldsymbol{Y})$.

Proof. Without loss of generality, for any nonshrink transformation τ where the new transformed point $\boldsymbol{y}^*$ was accepted, let y_i be a component of $\boldsymbol{y}^*$ where $y_i > u_i$ or $y_i < l_i$ and $y_{i'} = \phi(y_i)$. Let $\psi < \tau$ be the number such that $\phi(y_i) = c_i + \psi(c_i - y_n)$, then following Lagarias et al. [19], point $\boldsymbol{y}^*$ can be written in the following way,

$$\begin{aligned} \boldsymbol{y}^* = \boldsymbol{M}_k\boldsymbol{w}, \qquad & \text{where:} \\ & \boldsymbol{w} = \left(\frac{1+\tau}{n}, \dots, \frac{1+\psi}{n} \dots, \frac{1+\tau}{n}\right)^\top \\ & \boldsymbol{M}_k = [\boldsymbol{y}_0 - \boldsymbol{y}_n \quad \boldsymbol{y}_1 - \boldsymbol{y}_n \quad \cdots \quad \boldsymbol{y}_{n-1} - \boldsymbol{y}_n] \end{aligned} \tag{5.8}$$

assuming invariance of simplex ordering to the calculation of (2.11), for the volume of the new simplex $\boldsymbol{Y}_{k+1}$ using $\boldsymbol{M}_{k+1}$ instead of $\mathrm{L}(\boldsymbol{Y}_{k+1})$ we have the following computation,

$$|\det(\boldsymbol{M}_{k+1})| = |\det(\boldsymbol{M}_k - \boldsymbol{M}_k\boldsymbol{w}\boldsymbol{e}^\top)| = |\det(\boldsymbol{M}_k)| \cdot |\det(\boldsymbol{I} - \boldsymbol{w}\boldsymbol{e}^\top)|. \tag{5.9}$$

where $\boldsymbol{e} = (1, \dots, 1)^\top$. Let ψ be written as a difference of τ such that $\psi = \tau - r$. Matrix $\boldsymbol{I} - \boldsymbol{w}\boldsymbol{e}^\top$ has $n-1$ unitary eigenvalues and one which will be $1 - \boldsymbol{w}^\top\boldsymbol{e} = -(\tau - \frac{r}{n}) < -\tau$, contradicting Lemma 2.2.1. □

Along with the incorporation of (5.6) in the formulation of the problem, additional procedures to monitor the geometry of the solution simplex in a deterministic manner are presented in the hopes to make the algorithm invariant of starting point. In what follows, we propose three new procedures to the Nelder-Mead to solve nonlinear problems in the form of (5.1). The inclusion of the procedures result in a novel algorithm called the constrained Nelder Mead (c-NM). The procedures are: a uniform centered translated simplex for termination tests and efficient simplex restarts; a reinforced internal contraction step with safeguards that preserves volume to prevent shrinks; and a modified shrink step to merge two solution simplexes establishing an upper bound on the volume of the original simplex before the operation. We assume that the proposed alterations have access to a record $\boldsymbol{R}$ of the points that had the best objective function value f of each iteration where feasible solutions are always preferred over infeasible solutions. Algorithm 6 describes the steps of the constrained Nelder-Mead (c-NM).

5.4.1 The Uniform Translated Simplex Initialization

The uniform simplex $\mathbb{U}_n$ introduced in Section 2.2.2 can be a viable method to initialize a nondegenerate simplex in $\mathbb{R}^n$ due to the bounds on the simplex volume derived by the authors [42] and due to the fact that $\mathbb{U}_n$ retains its properties if it is scaled by any nonsingular $n \times n$ matrix $\boldsymbol{T}$. However, $\mathbb{U}$ cannot be freely re-scaled to an arbitrary interval $[a, b]$ because $\mathbb{U}$ would lose its properties if translated. This way, the starting simplex will always be centered in the origin and the vertices will either range between $[-1, a]$ or $[-b, 1]$. That would result in several problematic scenarios such as if the interior (int($\boldsymbol{Y}$)) of the simplex is infeasible, so any geometric transformation would yield an infeasible point; or if an entire orthant is infeasible.

We argue that if it is possible to somehow translate $\mathbb{U}_n$, then it is possible to center the simplex at any arbitrary centroid $\boldsymbol{c}$ and re-scale each row of $\mathbb{U}_n$ so that they can lie within problem bounds and consequently have a chance to be in the feasible region. We define the translation operation of the canonical uniform simplex $\mathbb{U}_n$ as a multiplication by a transformation matrix as follows,

Algorithm 6: Constrained Nelder-Mead (c-NM) algorithm

```
   Input: $f(\cdot)$, $\boldsymbol{x}_0$, $t$, $t'$, $\delta^{ic}$, $\delta^{oc}$, $\delta^{r}$, $\delta^{e}$
   Output: $\boldsymbol{Y}_t = \{\boldsymbol{y^0}, \boldsymbol{y^1}, \cdots, \boldsymbol{y^n}\}$ where $\boldsymbol{y^0} = \text{argmin}\, f(\boldsymbol{y}),\ \boldsymbol{y} \in \boldsymbol{Y}_t$
   Initialization: Initialize scaled uniform simplex $\boldsymbol{Y}_0 = \mathbb{U}_s$
1  for $i \leftarrow 1$ to $t$ do
2      if $\boldsymbol{Y}_i$ fails termination test then
3          if $\boldsymbol{y}^0$ is feasible then
4              if $\frac{1}{t'} \sum_{i=t'}^{n} \boldsymbol{r}^i = \boldsymbol{y}^0$ $\boldsymbol{r} \in \boldsymbol{R}$ then
5                  Restart $\boldsymbol{Y}_i$ building $\mathbb{U}_t$ from $\boldsymbol{r'} = \{\boldsymbol{r} \in \boldsymbol{R} \mid h(\boldsymbol{r}) + g(\boldsymbol{r}) = 0\}$
6              else
7                  Restart $\boldsymbol{Y}_i$ building $\mathbb{U}_t$ from
                     $\boldsymbol{r}^* = \{\boldsymbol{r} \in \boldsymbol{R} \mid \ \min f(\boldsymbol{r})$ and $h(\boldsymbol{r}) + g(\boldsymbol{r}) = 0\}$
8          else
9              Restart $\boldsymbol{Y}_i$ building $\mathbb{U}_t$ from $\boldsymbol{x} = \mathcal{U}(l, u)$
10     else
11         $\boldsymbol{Y}_i \leftarrow \boldsymbol{Y}_{i-1}$
12     $\boldsymbol{Y}_i \leftarrow$ Order $(\boldsymbol{Y}_i)$
13     $\boldsymbol{c} \leftarrow$ ComputeCentroid $(\boldsymbol{Y}_i)$
14     $\boldsymbol{y}^r \leftarrow$ Reflection $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^r)$
15     Expansion $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^e)$
16     OutsideContraction $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^{oc})$
17     InsideContraction $(\boldsymbol{Y}_i, \boldsymbol{c}, \delta^{ic})$
18     if $\boldsymbol{y}^c$ fails internal contraction then
19         $\boldsymbol{y}^{in} \leftarrow$ safeguardContraction$(\boldsymbol{y}_i, \boldsymbol{c}, \boldsymbol{y}^r)$
20         if $f(\boldsymbol{y}^{in}) > f(\boldsymbol{y}^c)$ then
21             $\boldsymbol{Y}_i \leftarrow$ merge$(\boldsymbol{Y}_i, \Delta)$
22 end
23 return $\boldsymbol{Y}_t$
```

$$\mathbb{U}_t = \boldsymbol{T}\mathbb{U}'_n$$

$$\text{where} \quad \boldsymbol{T} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & c_1 \\ 0 & 1 & 0 & \cdots & 0 & c_2 \\ 0 & 0 & 1 & \cdots & 0 & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & c_n \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbb{U}'_n = \begin{bmatrix} \mathbb{U} \\ 1 \end{bmatrix} \tag{5.10}$$

Where the last column of $\mathbb{U}_t$ is disregarded. Although properties (iv), (v) and (vi) explained in Section 2.2.2 are lost, $\mathbb{U}_t$ is still a nondegenerate simplex, as shown in Proposition 5.4.2.

Proposition 5.4.2. Translated uniform simplex $\mathbb{U}_t$ is a nondegenerate simplex.

Proof. Let $\mathbb{U}_t = [\boldsymbol{M}_t|y_t^0]$ and $\mathbb{U}_n = [\boldsymbol{M}|y^0]$. After the translation, diag$(\boldsymbol{M}_t) =$ diag$(\boldsymbol{M}) + \boldsymbol{c}$, where each component of diag$(\boldsymbol{M}_t)$ is distinct from each other. Since $y_t^0 \neq y_t^i \; i = 1, \dots, n$ then diag(L$(\mathbb{U}_t)) \neq 0$. From the definition (2.11) of simplex volume, $|\det(\text{L}(\mathbb{U}_t))| \neq 0$, characterizing $\mathbb{U}_t$ as a nondegenerate simplex. □

Uniform simplex $\mathbb{U}_n$ can now be rescaled to be within any feasible interval $[c, d]$ by a translation operation in the form of (5.10) followed by a scaling operation, i.e., multiplying by a diagonal matrix $\boldsymbol{Z}$. Given that $\mathbb{U}_n$ is within the $[a, b]$ interval, a scaled translated simplex $\mathbb{U}_s$ can be achieved in the following way:

$$\mathbb{U}_s = \boldsymbol{Z}\mathbb{U}_t$$

$$\text{where} \quad \boldsymbol{T} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & t_1 \\ 0 & 1 & 0 & \cdots & 0 & t_2 \\ 0 & 0 & 1 & \cdots & 0 & t_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & t_n \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{Z} = \begin{bmatrix} z_1 & 0 & 0 & 0 & 0 \\ 0 & z_2 & 0 & 0 & 0 \\ 0 & 0 & z_3 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & z_n \end{bmatrix} \tag{5.11}$$

$$t_i = c_i(b_i - 2a_i) - a_i d_i, \qquad z_i = \frac{d_i - c_i}{b_i - a_i} \quad i = 1, \dots, n$$

Clearly, evidence that $\mathbb{U}_s$ is a nondegenerate simplex, i.e., vol$(\mathbb{U}_s)$ follows from Proposition 5.4.2. Furthermore, $\mathbb{U}_s$ may be scaled by a scalar number, column-wise by a vector or by a $n \times n$ matrix.

We suggest initialization with $\mathbb{U}_s$ instead of $\mathbb{U}_t$ because $\mathbb{U}_s$ is roughly located in the middle of the search space, and if properly scaled results in a simplex that covers a large region, which according to Wessing [40] increases the possibility of the solution set to converge towards an accumulation point. On the other hand, we use $\mathbb{U}_t$ to restart the NM if it fails any termination test, choosing a new centering point $\boldsymbol{c}$ depending on the outcome of the three following cases:

1. No feasible solution has been found so far.
2. A feasible solution has been found but the simplex terminated because the simplex has degenerated or converged to a point.

3. A feasible solution has been found but the simplex collapsed into a subspace.

For case 1, a new centering location $\boldsymbol{c}$ for a new simplex $\mathbb{U}_t$ can either be obtained by a point sampled under any random distribution, or chosen from the best infeasible point in the record $\boldsymbol{R}$. In case 2, the point can be chosen from the best feasible point found so far. Lastly, in case 3, any feasible point may be chosen, preferably one that has euclidean distance less than ϵ to the best feasible point so far.

5.4.2 Safeguards for internal contractions and enhanced shrinks

Various works, e.g., [19, 30] have assessed the efficacy of the shrink operation, and how often it occurs. Torczon [30] showed that not only shrinks are extremely rare, reporting only 33 instances of shrink in 2.9 million iterations, but they also deteriorate the volume of the simplex [31, 33]. This way, it is common to find implementations where the shrink step is omitted in favor of stronger descent condition or termination test that relies on simplex volume.

The aforementioned works provided a good foundation to the understanding of the algorithm, but it was only validated for unconstrained convex functions. An empirical investigation was conducted to assess the frequency of the shrinks steps in constrained $\mathcal{C}^0$ functions. As a pilot study, we ran the Nelder-Mead 1000 times using the augmented lagrangian penalty method (5.6) and the $\mathbb{U}_s$ as starting simplex to solve the Himmelblau nonlinear function [86], a well-known nonlinear benchmark function, and recorded the number of the geometrical transformation steps that were accepted at each run. The Himmelblau nonlinear function has the following formulation,

$$\begin{aligned}
\underset{\boldsymbol{x}\in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = 5.3578547x_3^2 + 0.8356891x_1x_5 \\
\text{subject to} \quad & g_1(\boldsymbol{x}) = 85.334407 + 0.0056858x_2x_5 + Vx_1x_4 - 0.0022053x_3x_5, \\
& g_2(\boldsymbol{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 - 0.0021813x_3^2, \\
& g_3(\boldsymbol{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 - 0.0019085x_3x_4, \\
& V = 0.0006262
\end{aligned} \tag{5.12}$$

Each constraint is bounded by the following intervals: $0 \leq g_1(\boldsymbol{x}) \leq 92; 90 \leq g_2(\boldsymbol{x}) \leq 110; 20 \leq g_3(\boldsymbol{x}) \leq 25$. Decision variables $\{x_1, x_2, x_3, x_4, x_5\}$ feature the following bounds: $78 \leq x_1 \leq 102; 33 \leq x_2 \leq 45; 27 \leq \{x_3, x_4, x_5\} \leq 45$. Table 5.1 shows the mean number of each accepted step of transformations for the 1000 runs.

Operation	**Mean**
Reflection	2184
Expansion	220
Contraction	2837
Shrink	132

Table 5.1: Mean number of accepted steps of 1000 runs of the Himmelblau nonlinear function

The results of the experiment corroborates that shrinks are indeed rare (2% of the time) but they cannot be considered to be negligible and therefore must be stay in the implementation of the Nelder-Mead.

Another important point is that we have observed that around 95% of all contraction steps were inside contractions, due to either a large initial simplex or to the interior of the simplex be inside an infeasible region. Therefore, we propose an inclusion of a safeguard for the failed internal contraction case. If an internal contraction fails, then a simple test is carried to verify if a point further to the interior of the simplex is feasible and/or better than the contracted point. The test consists of doing another internal contraction from the contracted point and the worst vertex,

$$\boldsymbol{y}^{in} = \boldsymbol{y}^{ic} + \delta^{ic}(\boldsymbol{y}^{ic} - \boldsymbol{y}^{n}). \tag{5.13}$$

If $\boldsymbol{y}^{ic}$ is infeasible while $\boldsymbol{y}^{in}$ is feasible then we can claim that simplex $\boldsymbol{Y_k}$ is still able to converge to a feasible accumulation point in its interior. The step is accepted, $\boldsymbol{y}^{in}$ is accepted and the iteration terminates with $\boldsymbol{Y}_{k+1} = [\boldsymbol{y}^0\ \boldsymbol{y}^1\ \cdots\ \boldsymbol{y}^{in}]$. Otherwise, if $f(\boldsymbol{y}^{ic}) < f(\boldsymbol{y}^{in})$, we assume that simplex $\boldsymbol{Y}_{k+1}$ is better off moved elsewhere rather than being shrunk. In this situation, a merging of simplex is performed in which $\boldsymbol{Y}_k$ merges with simplex $\boldsymbol{Y}_s$, where $\boldsymbol{Y}_s \in \text{int}(\boldsymbol{Y}_k)$ so that $\text{vol}(\boldsymbol{Y}_{k+1}) = \tau\text{vol}(\boldsymbol{Y}_k),\ \tau < 1$. Construction of $\boldsymbol{Y}_s$ can be made using translated uniform simplex $\mathbb{U}_s$ centering at the centroid of $\boldsymbol{Y}_k$, enforcing that the maximum distance from vertices to $\boldsymbol{c}$ to be smaller than the oriented length σ^+ of $\boldsymbol{Y}_k$. Given two arbitrary simplices $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$, we define the merge operation from simplex $\boldsymbol{Y}_1$ resulting in a new simplex $\boldsymbol{Y}_n$ as,

$$\boldsymbol{Y}_n = \{\boldsymbol{y}_1^0, \boldsymbol{y}_1^1 + \gamma^m(\boldsymbol{y}_2^0 - \boldsymbol{y}_1^1), \boldsymbol{y}_1^2 + \gamma^m(\boldsymbol{y}_1^0 - \boldsymbol{y}_2^2), \boldsymbol{y}_1^n + \gamma^m(\boldsymbol{y}_2^{n-1} - \boldsymbol{y}_1^n)\}. \tag{5.14}$$

Where γ^m is a step length parameter empirically suggested to be 0.8. Proposition 5.4.3 shows that the simplex $\boldsymbol{Y}_n$ made by the merging operation is a nondegenerate simplex.

Proposition 5.4.3. Merged simplex $\boldsymbol{Y}_n$ is a nondegenerate simplex, i.e., $\text{vol}(\boldsymbol{Y}_n) > 0$.

Proof. If $\boldsymbol{Y}_1 = \boldsymbol{Y}_2$, then $\det(|\text{L}(\boldsymbol{Y}_n)|) = \det(|\text{L}(\boldsymbol{Y}_1)|) \neq 0$, so $\text{vol}(\boldsymbol{Y}_n)) > 0$. Otherwise, let $\text{L}(\boldsymbol{Y}_n) = |\text{L}(\boldsymbol{Y}_1) + \boldsymbol{D}|$ where $\boldsymbol{D} = [\gamma^m(\boldsymbol{y}_2^0 - \boldsymbol{y}_1^1)\ \cdots\ \gamma^m(\boldsymbol{y}_2^{n-1} - \boldsymbol{y}_1^n)]^\top$. So, $\text{L}(\boldsymbol{Y}_1)_i + \boldsymbol{D}_i \neq 0$ for $i = 1, \ldots, n-1$. Suppose there is a $\text{L}(\boldsymbol{Y}_n)_k$ such that $\text{L}(\boldsymbol{Y}_1)_k + \boldsymbol{D}_k = 0$, then $\text{sgn}(\text{L}(\boldsymbol{Y}_1)_k) \neq \text{sgn}(\boldsymbol{D}_k)$. However, for that to happen, $\text{sgn}(\boldsymbol{y}_1^i)$ in $\text{L}(\boldsymbol{Y}_1)_k$ has to be the opposite of $\text{sgn}(\boldsymbol{y}_1^i)$ in $\boldsymbol{D}_k$ which is not possible. $\square$

Ratio of the volume of the new merged simplex in relation to the first is defined in Proposition 5.4.4.

Proposition 5.4.4. The volume of merged simplex $\boldsymbol{Y}_n$ is $\kappa\text{vol}(\boldsymbol{Y}_1)$ where $\kappa = ((\gamma^m)^n \prod_{i=1}^n(\boldsymbol{y}_2^{i-1} - \boldsymbol{y}_1^i))^n$.

Proof. Without loss of generality, $f(\boldsymbol{y}_1^0) < f(\boldsymbol{y}_2^0) - \epsilon_1$ and $\|\boldsymbol{c_1} - \boldsymbol{c_2}\| \leq \epsilon_2$. Let $\text{L}(\boldsymbol{Y}_n) = [\boldsymbol{y}_1^1 - \boldsymbol{y}_0^1 + \gamma^m(\boldsymbol{y}_2^0 - \boldsymbol{y}_1^1)\ \cdots\ \boldsymbol{y}_1^n - \boldsymbol{y}_0^1 + \gamma^m(\boldsymbol{y}_2^{n-1} - \boldsymbol{y}_1^n)]$. $\text{L}(\boldsymbol{Y}_n)$ can be rearranged in the following way,

$$\begin{aligned}
\mathrm{L}(\boldsymbol{Y}_n) &= \gamma^m[(\boldsymbol{y}_2^0 - \boldsymbol{y}_1^1)\ \cdots\ (\boldsymbol{y}_2^{n-1} - \boldsymbol{y}_1^n)][(\boldsymbol{y}_1^1 - \boldsymbol{y}_1^0)\ \cdots\ (\boldsymbol{y}_1^n - \boldsymbol{y}_1^0)], \\
&= (\gamma^m)^n \prod_{i=1}^{n} (\boldsymbol{y}_2^{i-1} - \boldsymbol{y}_1^i))[(\boldsymbol{y}_1^1 - \boldsymbol{y}_1^0)\ \cdots\ (\boldsymbol{y}_1^n - \boldsymbol{y}_1^0)], \\
&= (\gamma^m)^n \prod_{i=1}^{n} (\boldsymbol{y}_2^{i-1} - \boldsymbol{y}_1^i))\mathrm{L}(\boldsymbol{Y}_1), \\
&= \kappa \mathrm{L}(\boldsymbol{Y}_1).
\end{aligned} \tag{5.15}$$

Since $|\det(\kappa\mathrm{L}(\boldsymbol{Y}_1))| = \kappa^n|\det(\mathrm{L}(\boldsymbol{Y}_1))|$. From the definition of volume (2.11), we have that $\mathrm{vol}(\boldsymbol{Y}_n) = \kappa^n \mathrm{vol}(\boldsymbol{Y}_1)$. $\square$

This way, we determine simplex $\boldsymbol{Y}_2$ to be either $\boldsymbol{Y}_2 \subset \boldsymbol{Y}_1$ or $\boldsymbol{Y}_1 \cap \boldsymbol{Y}_2 \neq \emptyset$. Regardless if the simplex was moved elsewhere or merged, the iteration is terminated.

5.5 Numerical Experiment

A numerical experiment is conducted on 8 real life cases of constrained problems from the field of engineering to assess the robustness and performance of the proposed approach to the algorithms. The intent of this experiment is to answer the following research question: "Can the integration of the augmented Lagrangian to heuristics that uses a solution sets result in an improvement to their performance to solve constrained multimodal optimization problems?". Moreover, two more minor hypothesis follow the research question: "Given adjustments, can a reasonably simple direct descent method such as the Nelder-Mead be competitive against complex population based heuristics that are best suited to handle constrained engineering design problems?" and "Given sufficient adjustments, is the A-DVM capable of improving the search capabilities of ABC-based algorithms to a competitive degree?".

Chosen baseline of comparison were population based heuristics due to them hp;ding one of the best results in these instances, as seen in the literature. On a later moment, we compare the performance of the best algorithms of each approach against the best results found in the literature from a vast range of derivative-free algorithms. In the first moment, the proposed algorithms are compared against several well-known population based heuristics and their respective improved variants. The methods in question are: the original ABC from Karaboga [59]; a hybridization of the ABC with evolution strategies by Mollinetti et al. [87] (ABC+ES); a modification of the ABC from [6] for multimodal problems (ABC-Xm1 and ABC-Xm5); Particle Swarm Optimization from Kennedy and Eberhart [61]; Evolutionary Particle Swarm Optimization by Miranda and Fonseca [62]; Maximum Search Limitation Evolutionary Particle Swarm Optimization by Neto et al. [82] (MS-EPSO); Differential Evolution (DE) [63]; Quantum behaved Particle Swarm Optimization (QPSO) by Sun et al. [83].

The budget of function evaluations for each algorithm was set to 10^5 function evaluations (FE's). The population size to all population heuristics is fixed at 50. For PSO and QPSO, the inertia factor (w_1) was set to 0.6 and both cognitive and social parameters (w_2, w_3) to 1.8. For EPSO, the number of replicas is set to 1, communication and mutation ratio to 0.8. For Differential

Evolution (DE) [63] with *best1bin* strategy, F value was 0.5 and *CR* 0.9. For each version of the ABC: $Lit = SN \cdot n$. ABC-X parameters were $Lit = 1.06 \cdot n$, maximum population of 66 and minimum of 15 for ABC-Xm1 and $Lit = 0.83 \cdot n$. With exception of the Nelder-Mead, the rest use a barrier method in the form of (5.7) and a quadratic penalty function. The population size of all heuristics was set to 50.

The c-NM was initialized using the scaled centralized simplex $\mathbb{U}_n$, where each time the solution set is flagged to be restarted, it can only be restart if at least more than 5% of the maximum budget has been spent. We define the termination tests for restarting the solution set as the following:

$$\begin{aligned} &1. \quad \frac{1}{t'}\sum_{i=t'}^{n} \boldsymbol{r}^i \leq \epsilon_1 \qquad \boldsymbol{r}^i \in \boldsymbol{R} \\ &2. \quad 2 \cdot \frac{\left|f(\boldsymbol{y}^n) - f(\boldsymbol{y}^0)\right|}{\|f(\boldsymbol{y}^n)\| + \|f(\boldsymbol{y}^0)\|} \leq \epsilon_2 \\ &3. \quad \max_{j \neq l} \frac{\left|\boldsymbol{y}^j - \boldsymbol{y}^l\right|_1}{\max\{1, |\boldsymbol{y}^l|_1\}} \leq \epsilon_3. \end{aligned} \tag{5.16}$$

Where $t' = 50$, records in $\boldsymbol{R}$ are $\epsilon_1 = 10^{-4}$; ϵ_2 and ϵ_3 are set to 10^{-7}. Test 1 checks if the solution set has converged to a local minimizer by verifying the entries in the record $\boldsymbol{R}$, test 2 verifies if the size of the simplex that represents the solution set is small enough, and test 3 verifies if the simplex collapsed to a subspace.

The experiment was conducted in a machine with the following hardware configuration: Intel core i7-6700 "Skylake" 3.4 GHz CPU; 16 GB RAM DDR4 3200 clocked at 3000 MHz. The running operating system (OS) is UbuntuOS 18.04. All algorithms were written in the python 3 programming language. Floating point operations were handled by the numpy package, version 1.19.1.

Every problem instance is a problem in the form of (5.1) and each is explained in detail in the following sections.

5.5.1 Three Bar Truss Design Problem: TBT

The optimization of a two-dimensional function ($\boldsymbol{x_j} = \{x_1, x_2\}$) which represents the volume subject to stress constraints on each side of the lattice [88, 89, 90] is given as follow:

$$\begin{aligned} \underset{\boldsymbol{x} \in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = \left(2\sqrt{2}x_1 + x_2\right) \times \mathrm{l}. \\ \text{subject to} \quad & g_1(\boldsymbol{x}) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}\mathrm{P} - \sigma \leq 0, \\ & g_2(\boldsymbol{x}) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}\mathrm{P} - \sigma \leq 0, \\ & g_3(\boldsymbol{x}) = \frac{1}{x_1 + \sqrt{2}x_2}\mathrm{P} - \sigma \leq 0 \end{aligned} \tag{5.17}$$

The following box constraints: $0 \leq \{x_1,\ x_2\} \leq 1$ are then employed to bound the decision variables $\{x_1,\ x_2\}$, where l $=$ 100 cm, P $=$ 2 KN/cm^2, $\sigma =$ 2 KN/cm^2.

5.5.2 Design of a Pressure Vessel Problem (Variants 1 and 2): DPV1, DPV2

DPV1 and DPV2 features constraints and four design variables [91, 92, 93, 94, 95]. Which are: pressure vessel thickness ($T_s = x_1$), pressure vessel lid thickness ($T_h = x_2$), internal radius of the vessel ($R = x_3$) length of the vessel without the lid ($L = x_4$). Finally, the decision variables are: $\boldsymbol{x_j} = \{x_1,\ x_2,\ x_3,\ x_4\}$.

$$\begin{aligned} \underset{\boldsymbol{x}\in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3. \\ \text{subject to} \quad & g_1(\boldsymbol{x}) = -x_1 + 0.0193x_3 \leq 0, \\ & g_2(\boldsymbol{x}) = -x_2 + 0.00954x_3 \leq 0, \\ & g_3(\boldsymbol{x}) = -x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0, \\ & g_4(\boldsymbol{x}) = x_4 - 240 \leq 0. \end{aligned} \tag{5.18}$$

Where the bounds of the design variables $\{x_1, x_2, x_3, x_4\}$ of DPV1 are $1 \times 0.0625 \leq \{x_1, x_2\} \leq 99 \times 0.0625; 10 \leq \{x_3, x_4\} \leq 200$.

The upper bound of x_4 is expanded to 240 for the DPV2. Then, the bounds related to the design variables of DPV2 are set to $1 \times 0.0625 \leq \{x_1, x_2\} \leq 99 \times 0.0625; 10 \leq x_3 \leq 200; 10 \leq x_4 \leq 240$.

5.5.3 Speed Reducer Design with 11 Constraints: SRD-11

The SRD-11 was proposed by Golinski [96] and has as a primary objective the minimization of $f(\boldsymbol{x})$ subject to a speed reducer system [88].

This problem formulation consists of 11 constraints and 7 decision variables. Face width ($b = x_1$); teeth modulus ($m = x_2$); number of teeth in the pinion ($z = x_3$); length of first axis between bearing ($l_1 = x_4$); length of second axis between bearing ($l_2 = x_5$); diameter of the first axis ($d_1 = x_6$); diameter of the second ($d_2 = x_7$).

$$
\begin{aligned}
\underset{\boldsymbol{x}\in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = 0.785x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) \\
& - 1.508x_1(x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right) \\
\text{subject to} \quad & g_1(\boldsymbol{x}) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0 \\
& g_2(\boldsymbol{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0 \\
& g_3(\boldsymbol{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0, \\
& g_4(\boldsymbol{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0, \\
& g_5(\boldsymbol{x}) = \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 1.69\times 10^6}}{110x_6^3} - 1 \leq 0, \\
& g_6(\boldsymbol{x}) = \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5\times 10^6}}{85x_7^3} - 1 \leq 0, \\
& g_7(\boldsymbol{x}) = \frac{x_2x_3}{40} - 1 \leq 0, \\
& g_8(\boldsymbol{x}) = \frac{5x_2}{x_1} - 1 \leq 0, \\
& g_9(\boldsymbol{x}) = \frac{x_1}{12x_2} - 1 \leq 0, \\
& g_{10}(\boldsymbol{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0, \\
& g_{11}(\boldsymbol{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0.
\end{aligned} \tag{5.19}
$$

The bounds of the decision variables $\{x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6,\ x_7\}$ of the SRD11 are, $2.6 \leq x_1 \leq 3.6$; $0.7 \leq x_2 \leq 0.8$; $17 \leq x_3 \leq 28$; $7.3 \leq x_4 \leq 8.3$; $7.8 \leq x_5 \leq 8.3$; $2.9 \leq x_6 \leq 3.9$; $5.0 \leq x_7 \leq 5.5$, respectively.

5.5.4 Minimization of the Weight of Tension/Compression Spring - MWTCS

This problem is reported by Arora [97] and Belegundu [98] and consists of minimizing the weight of a tension/compression spring. The problem features four constraints representing the minimal deflection, shearing tension, surge frequency and external diameter limits. The design variables are the average spool diameter (x_1), yarn diameter (x_2), and active spool number (x_3).

$$\begin{aligned}
\underset{\boldsymbol{x}\in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = (x_3+2)\,x_2x_1^2 \\
\text{subject to} \quad & g_1(\boldsymbol{x}) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0, \\
& g_2(\boldsymbol{x}) = \frac{4x_2^2 - x_1x_2}{12566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} - 1 \le 0, \\
& g_3(\boldsymbol{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0, \\
& g_4(\boldsymbol{x}) = \frac{x_1+x_2}{1.5} - 1 \le 0.
\end{aligned} \tag{5.20}$$

The bounds of the design variables $\{x_1,\ x_2,\ x_3\}$ for MWTCS are $0.05 \le x_1 \le 2; 0.25 \le x_2 \le 1.3; 2 \le x_3 \le 15$.

5.5.5 Welded Beam Design (Versions 1 and 2): WBD1, WBD2

The WBD problem describes the task of computing the minimum cost of building a welded beam, which is subject to shear stress conditions (τ), bending stress on the beam (θ), bar buckling load (P_c), final beam deflection (δ), and side constraints. This problem consists of four design variables ($\boldsymbol{x}_j = \{x_1,\ x_2,\ x_3,\ x_4\}$): Weld thickness ($h = x_1$), weld joint length ($l = x_2$); beam width ($t = x_3$); beam thickness $b = x_4$. In this work, we use two variants of the WBD, the first (WBD1) presents six constraints while the second (WBD2) adds a seventh constraint followed by changes in deflection ($\delta(\boldsymbol{x})$), buckling calculations ($P_c(\boldsymbol{x})$) and polar moment of inertia ($J(\boldsymbol{x})$).

$$\begin{aligned}
\underset{\boldsymbol{x}\in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14+x_2). \\
\text{subject to} \quad & g_1(\boldsymbol{x}) = \tau(x) - \tau_{max} \le 0, \\
& g_2(\boldsymbol{x}) = \sigma(x) - \sigma_{max} \le 0, \\
& g_3(\boldsymbol{x}) = x_1 - x_4 \le 0, \\
& g_4(\boldsymbol{x}) = 0.125 - x_1 \le 0, \\
& g_5(\boldsymbol{x}) = \delta(x) - 0.25 \le 0, \\
& g_6(\boldsymbol{x}) = P - P_c(x) \le 0.
\end{aligned} \tag{5.21}$$

Constraint $g_7(\boldsymbol{x})$ is added to the WBD2:

$$g_7(\boldsymbol{x}) = 0.10471x_1^2 + 0.04811x_3x_4\,(14+x_2) - 5 \le 0. \tag{5.22}$$

Finally, we apply the following constraints to WBD1 and WBD2: Maximum allowable weld shear stress ($\tau_{max} = 13600psi$); Maximum normal allowable stress for beam material ($\sigma_{max} = 30000psi$); system load (P $= 6000lb$); substrate distance (L $= 14in$); Shear modulus of the beam material (G $= 12 \times 10^6psi$); Young's module (E $= 30 \times 10^6psi$).

The constraints of the WBD1/WBD2 are described as follows:

- Primary tension (τ_1):

$$\tau_1 = \text{P} \times \left(\sqrt{2}x_1x_2\right)^{-1}. \tag{5.23}$$

- Secondary tension (τ_2):

$$\tau_2 = [R(\boldsymbol{x})\,(\mathrm{P}\,(\mathrm{L} + (0.5x_2)))] \times J(\boldsymbol{x})^{-1}. \tag{5.24}$$

- Weld shearing tension ($\tau(\boldsymbol{x})$):

$$\tau(\boldsymbol{x}) = \sqrt{\tau_1^2(\boldsymbol{x}) + 2\tau_1(\boldsymbol{x})\tau_2(\boldsymbol{x})\,(x_2 \times (2R(\boldsymbol{x}))^{-1}) + \tau_2^2(\boldsymbol{x})}. \tag{5.25}$$

- Polar inertia moment ($J(\boldsymbol{x})$):

$$J(\boldsymbol{x}) = \begin{cases} 2\left\{\frac{x_1x_2}{\sqrt{2}}\left[\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\} & \text{if WBD1,} \\ 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1+x_3}{2}\right)^2\right]\right\} & \text{if WBD2.} \end{cases} \tag{5.26}$$

- Moment over weld configuration center of gravity (R):

$$R(\boldsymbol{x}) = \sqrt{0.25x_2^2 + [0.5(x_1 + x_3)]^2}. \tag{5.27}$$

- Beam normal tension($\sigma(\boldsymbol{x})$):

$$\sigma(\boldsymbol{x}) = 6\mathrm{PL} \times \left(x_4x_3^4\right)^{-1}. \tag{5.28}$$

- Beam end Deflection ($\delta(\boldsymbol{x})$):

$$\delta(\boldsymbol{x}) = \begin{cases} 4\mathrm{PL}^3 \times \left(\mathrm{E}x_3^3x_4\right)^{-1} & \text{if WBD1,} \\ 6\mathrm{PL}^3 \times \left(\mathrm{E}x_3^3x_4\right)^{-1} & \text{if WBD2.} \end{cases} \tag{5.29}$$

- Bar buckling load (P_c):

$$P_c(\boldsymbol{x}) = \begin{cases} \frac{4.013\sqrt{\frac{\mathrm{EG}x_3^2x_4^6}{36}}}{\mathrm{L}^2}\left(1 - \frac{x_3}{2\mathrm{L}}\sqrt{\frac{\mathrm{E}}{4\mathrm{G}}}\right) & \text{if WBD1,} \\ \frac{4.013\mathrm{E}\sqrt{\frac{x_3^2x_4^6}{36}}}{\mathrm{L}^2}\left(1 - \frac{x_3}{2\mathrm{L}}\sqrt{\frac{\mathrm{E}}{4\mathrm{G}}}\right) & \text{if WBD2.} \end{cases} \tag{5.30}$$

WBD1 and WBD2 present the following decision variables $\{x_1,\ x_2,\ x_3,\ x_4\}$ and their bounds are as follows: $0.1 \leq x_1 \leq 2; 0.1 \leq x_2 \leq 10; 0.1 \leq x_3 \leq 10; 0.1 \leq x_4 \leq 2$.

5.5.6 Gear Train Design - GTD

Originally introduced by Sandgren [91], the problem consists of finding the optimal number of teeth of the gearwheel such that it minimizes the gear ratio cost. The formulation is as follows,

$$\begin{aligned} \underset{\boldsymbol{x}\in S}{\text{minimize}} \quad & f(\boldsymbol{x}) = \left(\frac{1}{6.931} - \frac{x_1x_2}{x_3x_4}\right)^2. \\ \text{subject to} \quad & 12 \leq \boldsymbol{x} \geq 60, \\ & \frac{x_1x_2}{x_3x_4} \in \mathbb{Z}^+. \end{aligned} \tag{5.31}$$

5.6 Computational Results and Discussion

Results of the numerical experiment are shown in Tables 5.2. Statistics for the comparison are the mean, standard deviation, median, and best-worst results obtained from 30 runs with distinct random seeds. For each problem instance, the row in boldface indicates the algorithm whose mean was the lowest among the others. Verification of statistical relevance between data is performed using two distinct tests, Friedman test as a omnibus test to verify overall differences and the Mann-Whitney U test to assess pairwise differences. Confidence level α for both tests is set to 0.95. Results of the statistical tests are displayed in Tables 5.3, 5.4 and 5.5 where values in boldface indicate whose the p-value was lower than 0.05. Lower bound of the precision of decimals is set to 12 digits, where anything below the bound is rounded to 0.

Log scaled plots of the mean of the best solutions of each algorithm throughout the iterations are shown in Figures 5.2 to 5.5. Fluctuations in the objective values such as in Figure 5.4a are frequently seen in the case c-NM due to the initial solution set be in infeasible regions that yield lower objective function values than its feasible counterparts. Figure 5.1 show the percentage of the best mean of the runs for each algorithm and the rank of each algorithm for each problem.

We begin by discussing the first four groups of problems. In the TBT problem, statistical difference has been observed between the c-NM and all the other algorithms with exception of the ABCX-m1+A-DVM and vice-versa($p < 0.05$). Performance of the c-NM and the A-DVM based algorithms was better than most ABC-based algorithms, except the ABCX-m1. Statistical evidence showed that the proposed algorithms fared worse than the PSO-based algorithms, albeit by a small margin (0.003 from the best solution and 0.015 from the mean of solutions) compared to the best algorithm in this instance (PSO). Concerning the DPV1 instance, no statistical relevance is found in the pairwise comparison of the c-NM against the EPSO, MS-EPSO, ABCx-m1, ABCx-m5, QPSO and ABC-ES ($p > 0.05$). For the rest, it has performed better than the ABC but worse than the ABC and the PSO. We attribute the drop in performance due to the fact that a direct search method such as the c-NM is sensitive to the choice of starting point and the randomness of the choice behind it. On the other hand, we can observe the influence of a good enough starting point in the DPV2 instance, where the c-NM obtained strictly better mean and best values compared to the others, although the pairwise comparison point to no statistical difference against the EPSO, ABC and QPSO ($p > 0.05$), where the first two performed almost as good as the c-NM is this instance. Lastly, in the SRD11 instance, a harder problem with more decision variables and constraints, the c-NM performed strictly worse than the EPSO, MS-EPSO, ABC, ABCx-m1 and the ABCx-m5,

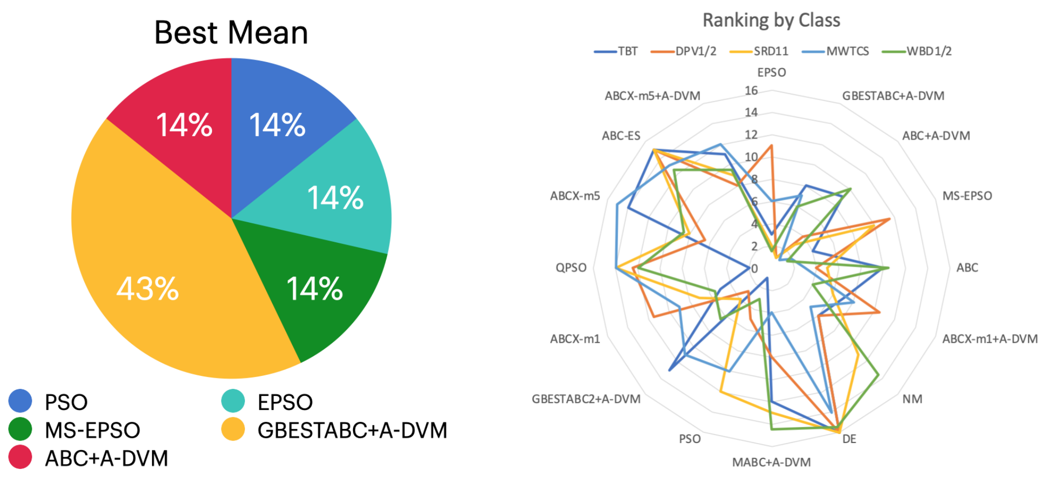
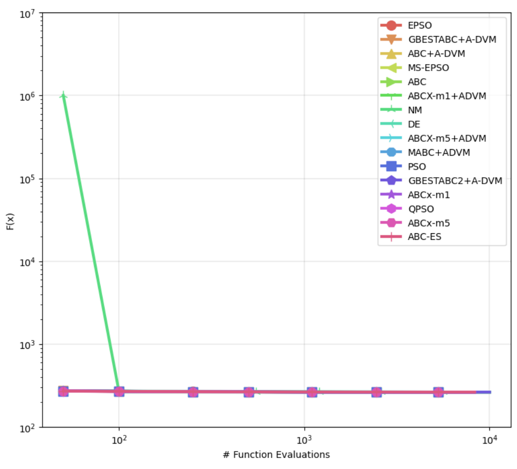


Figure 5.1: Percentage of the best overall mean and radar plot of the ranking of each technique according to the problem.

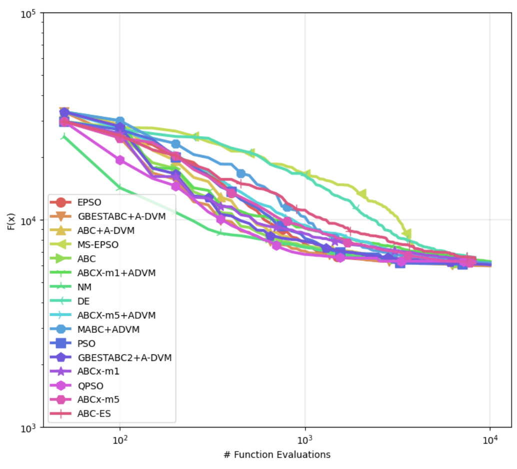
corroborated by the statistical evidence ($p < 0.05$). The poor performance can be attributed to two facts about the problem: The higher number of decision variables may have contributed to the deterioration of the performance; and the small interval of the decision variables may have hindered the construction of a initial simplex where at least one of its vertices is feasible. On the other hand, the A-DVM based approached achieved the lowest mean compared to all other approaches, backed by statistical evidence.

Now we focus into the last four problem sets. In the MWTCS, the c-NM instance was worse in comparison to the ABC, ABC+A-DVM, MABC+A-DVM and the MS-EPSO ($p < 0.05$) in terms of robustness. Although the mean value of the c-NM is higher than the two, it has achieved the lowest of the best values between them, what can again be explained by the algorithm sensitivity to starting points. In this instance, the algorithm with the lowest reported mean was the ABC+A-DVM. However, there statistical evidence does not support the assertion that the ABC+A-DVM had better performance than the second best algorithm, the MS-EPSO ($p > 0.05$).

In the next two instances, the WBD1 and WBD2 we can observe a similar trend from the MWTCS which reinforces the idea that the c-NM is indeed dependent on a starting point for the construction of an initial simplex. In the two instances, the robustness of the c-NM was not the best, but it has achieved the lowest best objective function value by a large margin compared to all other methods. Lastly, no statistical difference has been observed against all ($p > 0.05$) but the ABC-ES, whose values was close to 0 just like the rest. In the case of the A-DVM approaches, for the WBD1, the ABCX-m5+A-DVM had the second best mean among all other techniques, being worse than the MS-EPSO only ($p < 0.05$). In WBD2, we can observe the same thing, where the ABCX-m1+A-DVM holds the fourth best mean, bested by three PSO-based algorithms ($p < 0.05$). We can state that PSO-based algorithms perform better in instances in the likelihood of the WBD1 and WBD2. The GTD purpose in this experiment is to verify whether the c-NM and the A-DVM approaches were at least able to achieve the same performance as the others since the problem is not considered to be very problematic to solve compared to the rest, even with mixed integer constraints.



(a) TBT



(b) DPV1

Figure 5.2: Plots of the objective function of the TBT and DPV1 along the iterations

In summary, it was observed that the c-NM and the A-DVM based algorithms achieved competitive results in the DGT, MWTCS, TBT and WBD1 and WBD1 instances while achieving the best overall performance and robustness in the DPV2. Result of the experiment clearly indicates that the method is still highly dependent of a initial point to construct a simplex that is able to shrink towards a good accumulation point.

We now compare the best solution found by the c-NM and the A-DVM based algorithms against the best results found in the literature. The methods highlighted in the following tables are range from model-based derivative-free algorithms to parallel population heuristics.

5.7 Concluding Remarks

This chapter introduced several novel approaches for derivative-free methods to handle nonlinear constrained problems with nonlinear constraints focused on bringing deterministic features to algorithms that rely on randomness. First, we proposed an adaptation to the augmented lagrangian method to act as a penalty method to derivative-free algorithms that use a solution set in their optimization process. Then, using the augmented lagrangian, we presented a modified Nelder-Mead direct search (c-NM) and an A-DVM based Artificial Bee Colony to solve engineering design problems, a class of constrained optimization problems.

Several mechanisms were integrated to the c-NM so that the algorithm could handle constrained optimization problems while maintaining nondegeneracy of the geometry of the solution set: an extension of the uniform simplex initialization method that translates and scales the simplex freely so it can be used as a initialization and a restart method; safeguard steps for the internal step to avoid unnecessary shrinks towards infeasible regions; lastly, a merge step that replaces the shrink step in order to preserve a fraction of the volume of the original simplex. In the case of the A-DVM based ABC, along with the use of the augmented lagrangian, two new rules were added to the construction of the deterministic decision variable matrix $\boldsymbol{P}_d$. The rules take advantage of the problem

Problem	Algorithm	Mean	Median	Std. Dev	Best	Worst	Problem	Mean	Median	Std. Dev	Best	Worst
	EPSO	263.902	263.897	0.0185902	263.896	263.999		6258.62	6074.99	470.554	5804.39	7319
	GBESTABC+A-DVM	263.924	263.914	0.0225929	263.9	263.984		**5996.02**	**5968.83**	**130.14**	**5838.33**	**6478.85**
	ABC+A-DVM	263.929	263.921	0.0305446	263.898	264.026		6095.41	6080.28	133.858	5874.93	6338.14
	MS-EPSO	263.905	263.898	0.0147853	263.896	263.957		6297.57	6088.67	482.851	5804.76	7328.93
	ABC	263.934	263.92	0.0336655	263.898	264.009		6117.5	6094.12	144.908	5878.75	6557.77
	ABCX-m1+A-DVM	263.912	263.907	0.013249	263.897	263.942		6287.9	6280.41	239.269	5847.45	7047.52
	c-NM	263.911	263.91	0.0100473	263.899	263.941		6263.62	6203.62	325.158	5915.96	7580.9
TBT	DE	264.44	264.407	0.352103	263.979	265.548	DPV1	6581.98	6545.34	275.642	6049.94	7163.35
	ABCX-m5+A-DVM	263.956	263.93	0.0650091	263.898	264.187		6191.44	6169.7	152.145	5916.73	6591.66
	MABC+A-DVM	263.961	263.943	0.054714	263.897	264.137		6199.44	6184.74	158.393	5954.25	6565.34
	PSO	**263.896**	**263.896**	**0**	**263.896**	**263.896**		6111.65	5957.74	401.024	5804.38	7319
	GBESTABC2+A-DVM	263.956	263.95	0.0435331	263.903	264.071		6061.47	6062.03	100.085	5863.75	6345.44
	ABCx-m1	263.906	263.905	0.00571219	263.896	263.917		6333.65	6230.21	339.529	5880.05	7198.2
	QPSO	263.898	263.896	0.00496225	263.896	263.915		6311.08	6233.95	368.377	5804.64	6984.61
	ABCx-m5	264.019	263.995	0.10362	263.9	264.353		6185.45	6156.7	137.32	5987.34	6619.38
	ABC-ES	264.237	264.084	0.391611	263.903	265.669		6482.81	6327.1	616.308	5957.13	8803.45
	EPSO	6293.21	6191.67	386.574	5885.33	7299.97		2894.7	2894.38	1.69866	2894.38	2903.85
	GBESTABC+A-DVM	**6113.5**	**6050.65**	**182.972**	**5919.27**	**6827.4**		**2894.39**	**2894.38**	**0.00725378**	**2894.38**	**2894.42**
	ABC+A-DVM	6174.99	6141.61	144.884	5975.5	6656.55		2894.72	2894.62	0.326239	2894.41	2895.7
	MS-EPSO	6562.06	6510.42	502.333	5885.39	7319.26		2897.02	2894.4	4.97084	2894.38	2907.35
	ABC	6163.48	6151.27	123.422	5981.01	6463.47		2894.66	2894.55	0.255201	2894.42	2895.41
	ABCX-m1+A-DVM	6442.1	6433.74	249.129	6002.93	6953.12		2895.35	2895.31	0.537323	2894.59	2896.55
	c-NM	6155.07	6074.56	278.569	5822.05	7004.11		5493.69	3450.62	4056.27	3085.42	14992.5
DPV2	DE	6672.46	6644.77	252.526	6254.69	7223.92	SRD11	3.00288e+06	2903.5	4.58253e+06	2898.31	1.00028e+07
	ABCX-m5+A-DVM	6279.9	6261.19	148.558	6016.02	6578.2		2897.5	2897.3	1.6656	2895.13	2902.44
	MABC+A-DVM	6273.58	6281.43	150.672	6036.95	6711.52		3.00287e+06	2895.27	4.58253e+06	2894.4	1.00028e+07
	PSO	6191.57	5942.33	468.719	5885.33	7319		2.66955e+06	2907.8	4.42212e+06	2894.38	1.00028e+07
	GBESTABC2+A-DVM	6170.67	6136.46	129.76	5975.77	6520.31		2894.76	2894.68	0.303282	2894.44	2895.65
	ABCx-m1	6407.7	6369.1	364.241	5948.33	7251.69		2895.39	2895.32	0.63144	2894.54	2897.16
	QPSO	6361.7	6338.12	377.213	5901.42	7316.22		1.33622e+06	2894.39	3.39932e+06	2894.38	1.00028e+07
	ABCx-m5	6273.04	6242.29	173.368	6033.64	6710.54		2898.09	2897.67	1.80686	2895.35	2902.91
	ABC-ES	6610.37	6397.1	877.584	5984.53	11050.6		1.3363e+06	3019.2	3.39925e+06	2897.54	1.00027e+07
	EPSO	1e+06	0.0134979	3e+06	0.0126962	1e+07		2.41427	2.39666	0.0506303	2.38209	2.64702
	GBESTABC+A-DVM	1.33333e+06	0.013327	3.39935e+06	0.0128604	1e+07		2.6024	2.54558	0.16534	2.39355	2.9097
	ABC+A-DVM	**0.0137128**	**0.0134881**	**0.00090146**	**0.0129082**	**0.016592**		2.65941	2.62598	0.151932	2.44017	3.01794
	MS-EPSO	0.0142919	0.0135306	0.00181443	0.0126947	0.0178255		**2.40767**	**2.39981**	**0.0279734**	**2.38108**	**2.49301**
	ABC	0.0140586	0.0138035	0.00114824	0.0128934	0.0183524		2.73917	2.70998	0.206092	2.42926	3.43969
	ABCX-m1+A-DVM	2.66667e+06	0.0153538	4.42217e+06	0.0127314	1e+07		2.57514	2.52168	0.144333	2.39898	2.95884
	NM	0.0253572	0.0126886	0.0459053	0.0126665	0.236589		2.66087	2.51524	0.427917	2.33394	4.36388
MWTCS	DE	666667	0.0162031	2.49444e+06	0.0135812	1e+07	WBD1	2.87199	2.87147	0.23828	2.49377	3.63494
	ABCX-m5+A-DVM	1e+06	0.0134809	3e+06	0.0127627	1e+07		2.63372	2.58517	0.130541	2.45812	2.89238
	MABC+A-DVM	0.0144215	0.0138394	0.00154204	0.0129177	0.0198077		2.86376	2.79578	0.249814	2.45722	3.4028
	PSO	5e+06	5e+06	5e+06	0.0126652	1e+07		2.43888	2.4004	0.0656665	2.381	2.66988
	GBESTABC2+A-DVM	1e+06	0.0132133	3e+06	0.012677	1e+07		2.58832	2.55166	0.140864	2.41664	3.07058
	ABCx-m1	1e+06	0.0143452	3e+06	0.0128131	1e+07		2.57701	2.54986	0.13701	2.41927	2.90046
	QPSO	3.66667e+06	0.0144453	4.81894e+06	0.0126652	1e+07		2.74965	2.63239	0.430884	2.38112	4.10117
	ABCx-m5	1.33333e+06	0.014431	3.39935e+06	0.0129064	1e+07		2.65598	2.63346	0.14254	2.44803	3.00644
	ABC-ES	1.66667e+06	0.0175164	3.72678e+06	0.0128713	1e+07		2.79867	2.81571	0.208077	2.43841	3.26427
	EPSO	**2.39752**	**2.39124**	**0.0191024**	**2.38106**	**2.45111**		**0**	**0**	**0**	**0**	**0**
	GBESTABC+A-DVM	2.59167	2.525	0.181293	2.40472	3.05628		0	0	0	0	0
	ABC+A-DVM	2.72329	2.66188	0.241642	2.42432	3.30128		0	0	0	0	0
	MS-EPSO	2.43637	2.41441	0.0565839	2.38097	2.60242		0	0	0	0	0
	ABC	2.71395	2.68453	0.161994	2.41298	3.20774		0	0	0	0	0
	ABCX-m1+A-DVM	2.53591	2.51429	0.110458	2.41082	2.87406		0	0	0	0	0
	NM	3.4216	3.4927	1.62804	1.66504	8.62822		0	0	0	0	0
WBD2	DE	2.97089	2.91116	0.32575	2.46245	3.94647	DGT	0	0	0	0	0
	ABCX-m5+A-DVM	2.72949	2.6859	0.213448	2.44488	3.39382		0	0	0	0	0
	MABC+A-DVM	2.93744	2.95778	0.299431	2.42063	3.43781		0	0	0	0	0
	PSO	2.41396	2.39122	0.0437111	2.38096	2.55952		0	0	0	0	0
	GBESTABC2+A-DVM	2.65416	2.60279	0.164443	2.45798	3.18699		0	0	0	0	0
	ABCx-m1	2.60966	2.51004	0.242982	2.40439	3.31542		0	0	0	0	0
	QPSO	2.86493	2.5815	0.697748	2.38115	4.86203		0	0	0	0	0
	ABCx-m5	2.6785	2.66348	0.137934	2.4205	3.03166		0	0	0	0	0
	ABC-ES	2.86854	2.71453	0.344784	2.45195	3.65891		0	0	0	0	0

Table 5.2: Statistical results of all problems

TBT	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	c-NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	1.41572e-08	1.82295e-08	0.141889	8.48976e-09	9.30424e-07	1.62777e-07	1.66919e-11	6.01163e-09	1.59837e-09	6.0589e-13	7.05489e-10	1.79617e-05	0.0095562	1.57944e-10	6.02834e-11
GBESTABC+A-DVM	-	0.38656	1.06636e-05	0.181611	0.0146027	0.0135432	1.66919e-11	0.0287298	0.00118998	6.0589e-13	0.000650832	0.000327432	8.47362e-10	1.43949e-06	1.67598e-08
ABC+A-DVM	-	-	9.34084e-06	0.315438	0.0130387	0.00750707	2.25216e-11	0.0686614	0.00364418	6.0589e-13	0.00175058	8.91778e-05	1.0169e-09	5.09384e-06	3.54406e-08
MS-EPSO	-	-	-	3.36811e-06	0.00166931	0.000586879	1.50993e-11	5.9684e-07	3.00519e-08	6.0589e-13	1.09737e-08	0.0169371	0.00515734	8.47362e-10	1.7371e-10
ABC	-	-	-	-	0.00364418	0.00266104	2.48758e-11	0.138595	0.0169371	6.0589e-13	0.0130387	6.23853e-05	6.43519e-10	2.80364e-05	1.62777e-07
ABCX-m1+A-DVM	-	-	-	-	-	0.479366	1.50993e-11	0.000476037	1.54695e-06	6.0589e-13	2.59284e-07	0.0928834	1.98238e-08	7.79038e-09	4.87775e-10
c-NM	-	-	-	-	-	-	1.50993e-11	0.000428206	6.4302e-07	6.0589e-13	2.79995e-07	0.0202975	7.79038e-09	2.98365e-09	4.05068e-10
DE	-	-	-	-	-	-	-	1.91008e-10	1.57944e-10	6.0589e-13	8.06613e-11	1.50993e-11	1.50993e-11	1.54053e-08	0.00242801
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.210193	6.0589e-13	0.245891	4.44144e-06	3.36098e-10	0.00159148	7.64584e-06
MABC+A-DVM	-	-	-	-	-	-	-	-	-	6.0589e-13	0.491154	2.34282e-08	8.06613e-11	0.0081424	2.6325e-05
PSO	-	-	-	-	-	-	-	-	-	-	6.0589e-13	6.0589e-13	6.0589e-13	6.0589e-13	6.0589e-13
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	1.00761e-08	4.07637e-11	0.00882451	3.17802e-05
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	1.90263e-07	4.4455e-10	1.18573e-10
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	3.03288e-11	1.84486e-11
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00561388
DPV1	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	0.114115	0.461721	0.28461	0.497051	0.0726596	0.159152	0.00092874	0.218821	0.245891	0.0686614	0.364133	0.0726596	0.159152	0.227648	0.0210334
GBESTABC+A-DVM	-	0.00231856	0.0339344	0.000178192	3.02297e-07	1.33921e-06	1.43579e-10	2.21025e-06	5.5386e-07	0.315438	0.00333444	1.24566e-06	0.000249091	4.42055e-07	2.34282e-08
ABC+A-DVM	-	-	0.260072	0.336748	0.000310133	0.0125506	1.74856e-09	0.00918398	0.00882451	0.0611765	0.189518	0.00102617	0.0250601	0.0107531	1.79617e-05
MS-EPSO	-	-	-	0.409373	0.141889	0.232136	0.00159148	0.353086	0.289647	0.0202975	0.185539	0.114115	0.260072	0.353086	0.0350633
ABC	-	-	-	-	0.000476037	0.0225731	5.5117e-09	0.0195835	0.0217918	0.030726	0.0768336	0.00231856	0.0374135	0.0225731	0.000119424
ABCX-m1+A-DVM	-	-	-	-	-	0.138595	4.33172e-05	0.0543449	0.0559934	0.00102617	4.14597e-06	0.473478	0.420901	0.0287298	0.236673
NM	-	-	-	-	-	-	1.91533e-05	0.409373	0.39223	0.00380853	0.00107832	0.201769	0.397923	0.39223	0.0543449
DE	-	-	-	-	-	-	-	1.18841e-07	2.79995e-07	2.37225e-06	3.69014e-10	0.000586879	0.00192403	5.33284e-08	0.0039795
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.485258	0.00585534	0.000150294	0.0978954	0.201769	0.353086	0.0140643
MABC+A-DVM	-	-	-	-	-	-	-	-	-	0.00364418	0.000345626	0.105781	0.241258	0.39223	0.0259386
PSO	-	-	-	-	-	-	-	-	-	-	0.0747244	0.00113289	0.00636606	0.00515734	0.000134029
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	3.59939e-05	0.0095562	0.000134029	4.09875e-07
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	0.403637	0.0768336	0.227648
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	0.210193	0.218821
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00561388
DPV2	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	0.122907	0.420901	0.0202975	0.380914	0.0157328	0.170144	4.33172e-05	0.159152	0.155594	0.0225731	0.479366	0.0811875	0.155594	0.19355	0.00847744
GBESTABC+A-DVM	-	0.0125506	0.000451535	0.0146027	5.13867e-07	0.336748	4.4455e-10	1.29868e-05	2.31948e-05	0.0339344	0.0130387	0.000345626	0.00538131	5.52886e-05	5.13867e-07
ABC+A-DVM	-	-	0.0049417	0.467596	2.04198e-05	0.0747244	5.35089e-10	0.00348622	0.00291408	0.00561388	0.479366	0.0130387	0.0811875	0.00847744	1.68407e-05
MS-EPSO	-	-	-	0.00434219	0.255299	0.00137742	0.189518	0.0496288	0.0424998	0.000839878	0.00333444	0.19355	0.0768336	0.0452452	0.397923
ABC	-	-	-	-	8.73957e-06	0.0789878	1.43579e-10	0.00242801	0.00254211	0.00847744	0.444151	0.00720609	0.0726596	0.00847744	6.68338e-06
ABCX-m1+A-DVM	-	-	-	-	-	1.68407e-05	0.0012497	0.00515734	0.00364418	0.000100291	1.38629e-05	0.185539	0.0811875	0.00473413	0.461721
NM	-	-	-	-	-	-	1.54053e-08	0.000501766	0.000476037	0.0880638	0.0559934	0.00242801	0.0225731	0.00102617	1.06636e-05
DE	-	-	-	-	-	-	-	3.26387e-08	1.54053e-08	5.45343e-06	1.7371e-10	0.0012497	0.000327432	4.91645e-08	0.00473413
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.420901	0.000618092	0.00211296	0.210193	0.485258	0.364133	0.0095562
MABC+A-DVM	-	-	-	-	-	-	-	-	-	0.000721164	0.00291408	0.173914	0.409373	0.39223	0.0049417
PSO	-	-	-	-	-	-	-	-	-	-	0.00585534	0.000798443	0.00192403	0.00102617	7.92305e-05
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.0103404	0.0747244	0.00847744	9.34084e-06
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	0.279615	0.162763	0.159152
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	0.420901	0.066727
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00434219
SRD11	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	1.78429e-06	5.86249e-10	3.00266e-08	1.01586e-09	2.78328e-10	1.50804e-11	7.31299e-11	2.78328e-10	1.18438e-10	3.04554e-10	4.44074e-10	2.78328e-10	1.43857e-06	2.78328e-10	1.80289e-11
GBESTABC+A-DVM	-	1.84486e-11	0.000883282	1.66919e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.50993e-11	1.84486e-11	7.70451e-10	1.50993e-11	1.50993e-11	0.181611	1.50993e-11	1.47713e-11
ABC+A-DVM	-	-	0.00515734	0.145236	4.7666e-07	1.50993e-11	1.50993e-11	3.03288e-11	2.17654e-05	7.63126e-06	0.138595	1.54695e-06	0.000345626	1.84486e-11	1.47713e-11
MS-EPSO	-	-	-	0.0130387	0.000428206	1.50993e-11	1.39144e-07	7.46578e-05	1.21636e-05	7.4341e-07	0.00231856	0.000476037	0.105781	5.52886e-05	7.90352e-11
ABC	-	-	-	-	9.36549e-08	1.50993e-11	1.50993e-11	1.66919e-11	3.86934e-06	3.13489e-06	0.0250601	2.05635e-07	0.000501766	1.66919e-11	1.47713e-11
ABCX-m1+A-DVM	-	-	-	-	-	1.50993e-11	1.50993e-11	2.49898e-09	0.274663	4.59839e-05	2.21025e-06	0.491154	1.21636e-05	7.05489e-10	1.47713e-11
NM	-	-	-	-	-	-	0.0039795	1.50993e-11	0.0039795	0.000975346	1.50993e-11	1.50993e-11	5.5386e-07	1.50993e-11	1.32433e-06
DE	-	-	-	-	-	-	-	4.87775e-10	0.00211296	0.269735	1.50993e-11	1.50993e-11	9.30424e-07	1.54053e-08	0.0296521
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.0268426	0.000150091	2.74703e-11	1.30076e-08	3.86934e-06	0.111286	5.90517e-11
MABC+A-DVM	-	-	-	-	-	-	-	-	-	0.0789676	0.000112695	0.264891	9.34084e-06	0.0202975	0.00689275
PSO	-	-	-	-	-	-	-	-	-	-	1.68111e-05	4.59839e-05	6.23448e-06	0.000177956	0.0233138
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	9.34084e-06	0.000210875	1.84486e-11	1.47713e-11
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	1.57864e-05	1.11363e-09	1.47713e-11
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	3.86934e-06	5.89761e-07
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	7.17327e-11

Table 5.3: p-values of pairwise Mann-Whitney U test of the TBT, DPV1, DPV2 and SRD11 instances.

MWTCS	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	0.325893	0.415122	0.432492	0.173897	0.012328	5.84547e-07	0.000149684	0.170127	0.0953498	0.103994	0.412205	0.0844666	0.0952078	0.0306232	8.01682e-05
GBESTABC+A-DVM	-	0.358576	0.397909	0.0593831	0.0178487	1.58786e-07	1.14537e-06	0.392215	0.0135325	0.237695	0.320582	0.0494973	0.143346	0.0348998	1.85896e-05
ABC+A-DVM	-	-	0.497051	0.0768336	0.00808009	5.43319e-07	2.49776e-09	0.245891	0.0146027	0.0911392	0.375287	0.0452362	0.0720438	0.0195699	1.88855e-07
MS-EPSO	-	-	-	0.201769	0.00361014	1.51985e-06	9.33834e-06	0.111286	0.108508	0.0505757	0.386554	0.0559835	0.0642484	0.0233627	5.06505e-06
ABC	-	-	-	-	0.0240758	5.0403e-07	4.52975e-08	0.415128	0.236673	0.0986877	0.116981	0.236661	0.154852	0.0834036	1.90564e-06
ABCX-m1+A-DVM	-	-	-	-	-	5.03742e-08	0.105256	0.0372439	0.0684319	0.460596	0.0149064	0.131431	0.296891	0.283839	0.0638636
NM	-	-	-	-	-	-	1.25833e-07	2.01421e-07	4.33495e-07	1.69381e-06	8.47505e-07	2.17354e-07	5.04282e-06	1.07027e-07	4.39232e-08
DE	-	-	-	-	-	-	-	0.000210836	2.54482e-06	0.211803	2.30823e-05	0.000118948	0.112934	0.00689275	0.158964
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.197634	0.133449	0.218808	0.227636	0.116278	0.103071	0.000277068
MABC+A-DVM	-	-	-	-	-	-	-	-	-	0.0986877	0.0411701	0.444148	0.259432	0.227617	1.57066e-05
PSO	-	-	-	-	-	-	-	-	-	-	0.124214	0.200623	0.35847	0.281463	0.440117
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.073588	0.0878732	0.030625	5.94352e-05
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	0.354958	0.25991	0.000238631
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	0.449606	0.0970456
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00712474
WBD1	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	3.00519e-08	1.57944e-10	0.473478	8.06613e-11	2.73087e-09	8.40307e-05	2.74703e-11	1.30493e-10	3.34776e-11	0.10044	8.47362e-10	6.55552e-09	0.000134029	1.18573e-10	7.32153e-11
GBESTABC+A-DVM	-	0.0611765	7.79038e-09	0.00453438	0.409373	0.23213	7.14918e-06	0.108508	3.3825e-05	1.29499e-05	0.491154	0.380914	0.289647	0.0686614	0.000278055
ABC+A-DVM	-	-	3.03288e-11	0.0594087	0.00691581	0.0120765	0.000150294	0.28461	0.000476037	4.22032e-09	0.0242067	0.0151587	0.403637	0.497051	0.00348622
MS-EPSO	-	-	-	3.03288e-11	7.73261e-10	2.98387e-05	1.50993e-11	2.74703e-11	2.03858e-11	0.114079	1.57944e-10	7.73261e-10	6.62476e-05	2.74703e-11	3.69454e-11
ABC	-	-	-	-	0.00022296	0.00131178	0.00882451	0.0163255	0.0350633	4.0268e-10	0.000528777	0.000345626	0.12594	0.0543449	0.0904498
ABCX-m1+A-DVM	-	-	-	-	-	0.289642	3.26307e-07	0.0268426	8.64516e-07	2.53715e-06	0.320712	0.449998	0.152088	0.0081424	2.6325e-05
NM	-	-	-	-	-	-	4.60356e-05	0.0202947	3.38093e-05	0.00560654	0.173907	0.304999	0.100468	0.0130367	0.000405862
DE	-	-	-	-	-	-	-	1.91533e-05	0.426691	3.67013e-11	9.30424e-07	6.92626e-07	0.00333444	0.000106323	0.141889
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	7.03343e-05	5.02625e-09	0.052735	0.029714	0.403637	0.28461	0.000650832
MABC+A-DVM	-	-	-	-	-	-	-	-	-	9.72208e-11	1.7854e-06	1.91747e-06	0.00720609	0.000428206	0.232136
PSO	-	-	-	-	-	-	-	-	-	-	8.61376e-07	8.15097e-06	0.000248596	5.98099e-09	1.1074e-09
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.358594	0.170144	0.0195835	3.59939e-05
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	0.166427	0.0120784	2.31948e-05
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	0.426691	0.0217918
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.0039795
WBD2	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	2.09984e-10	2.48758e-11	0.000428206	2.74703e-11	1.30493e-10	0.0928834	1.50993e-11	2.25216e-11	2.25216e-11	0.269755	1.50993e-11	4.05068e-10	8.40658e-05	2.25216e-11	1.50993e-11
GBESTABC+A-DVM	-	0.00882451	3.36811e-06	0.00131214	0.241258	0.12594	7.45902e-07	0.00166931	5.83718e-06	5.33284e-08	0.0151587	0.38656	0.455854	0.00434219	0.000112695
ABC+A-DVM	-	-	3.88627e-09	0.342161	0.000557128	0.166427	0.00092874	0.315438	0.00348622	4.05068e-10	0.236673	0.00882451	0.132163	0.491154	0.0611765
MS-EPSO	-	-	-	3.06052e-10	7.14918e-06	0.0928834	6.64426e-11	3.06052e-10	2.09984e-10	0.0125506	1.21931e-09	1.38629e-05	0.00882451	4.87775e-10	3.06052e-10
ABC	-	-	-	-	5.09384e-06	0.170144	0.000249091	0.420901	0.00380853	6.64426e-11	0.0268426	0.00113289	0.0904498	0.218821	0.152088
ABCX-m1+A-DVM	-	-	-	-	-	0.108508	1.41572e-08	1.06636e-05	2.22202e-07	8.64705e-08	0.000384864	0.426691	0.260072	2.17654e-05	5.09384e-06
NM	-	-	-	-	-	-	0.294726	0.166427	0.241258	0.0928834	0.135353	0.132163	0.320712	0.166427	0.205955
DE	-	-	-	-	-	-	-	0.000528777	0.473478	3.34776e-11	2.04198e-05	5.83718e-06	0.00333444	3.59939e-05	0.0928834
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.00847744	8.88454e-11	0.0339344	0.00092874	0.0928834	0.331367	0.197634
MABC+A-DVM	-	-	-	-	-	-	-	-	-	4.95931e-11	0.000451535	1.57864e-05	0.00585534	0.00092874	0.177736
PSO	-	-	-	-	-	-	-	-	-	-	9.7839e-11	1.18841e-07	0.000178192	8.06613e-11	6.64426e-11
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.0103404	0.245891	0.114115	0.0095562
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	0.444151	0.0039795	0.000134029
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	0.155594	0.0259386
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.0629851
GTD	GBESTABC+A-DVM	ABC+A-DVM	MS-EPSO	ABC	ABCX-m1+A-DVM	NM	DE	ABCX-m5+A-DVM	MABC+A-DVM	PSO	GBESTABC2+A-DVM	ABCx-m1	QPSO	ABCx-m5	ABC-ES
EPSO	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
GBESTABC+A-DVM	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
ABC+A-DVM	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
MS-EPSO	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
ABC	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
ABCX-m1+A-DVM	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
NM	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
DE	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
ABCX-m5+A-DVM	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
MABC+A-DVM	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.18406	0.00550727
PSO	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.18406	0.00550727
GBESTABC2+A-DVM	-	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.18406	0.00550727
ABCx-m1	-	-	-	-	-	-	-	-	-	-	-	-	0.18406	0.18406	0.00550727
QPSO	-	-	-	-	-	-	-	-	-	-	-	-	-	0.18406	0.00550727
ABCx-m5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0.00550727

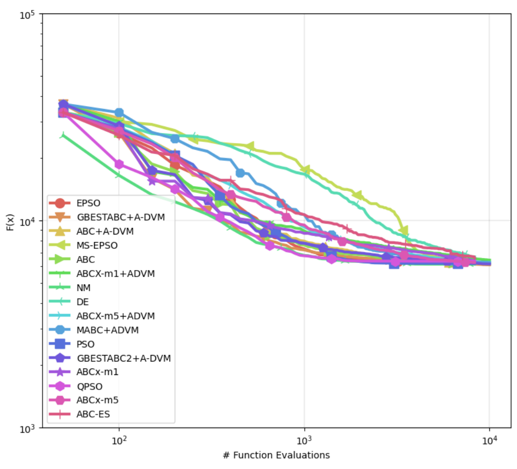
Table 5.4: p-values of pairwise Mann-Whitney U test of the MWTCS, WBD1, WBD2 and GTD instances

TBT	DPV1	DPV2	SRD11	MWTCS	WBD1	WBD2	GTD
1.168365e-43	**9.029975e-08**	**1.899573e-10**	**4.398067e-37**	**3.643795e-13**	**6.313696e-26**	**3.411428e-22**	**1.887828e-08**

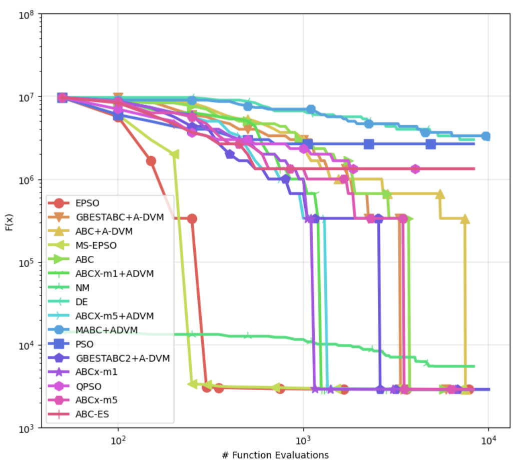
Table 5.5: p-values of Friedman test for all instances

Method	**Best**	**Mean**	**Worst**	**Std.Dev**	**Median**
Ray and Liew[99]	263.8958466	263.9033	263.96975	1.26×10^{-2}	263.8989
Zhang et al.[100]	263.8958434	263.8958436	263.8958498	9.72×10^{-7}	263.8958434
Garg[90]	263.8958433	263.8958437	263.8958459	5.34×10^{-7}	263.8958436
c-NM	263.899	263.911	263.941	0.0100473	263.91
ABCX-m1+A-DVM	263.897	263.912	263.942	0.013249	263.942

Table 5.6: Results of the TBT throughout the literature

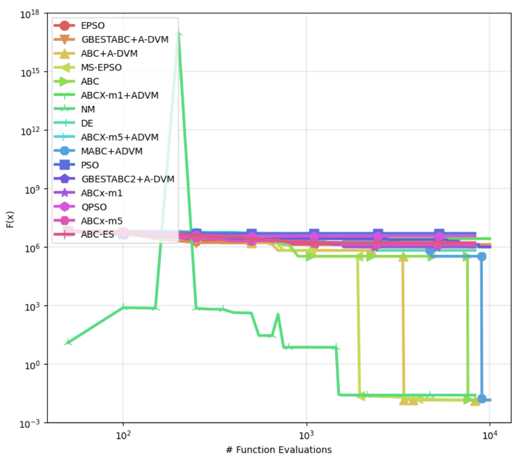


(a) DPV2

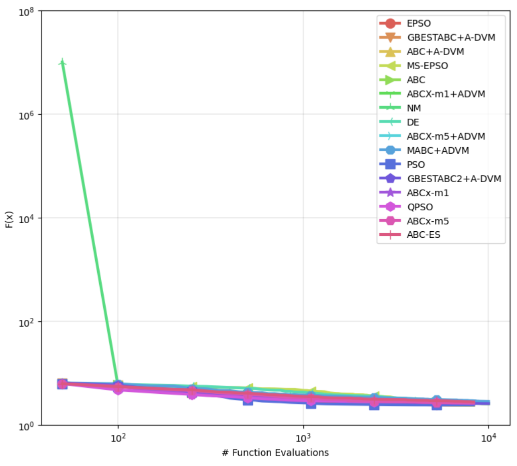


(b) SRD11

Figure 5.3: Plots of the objective function of the DPV2 and SRD11 along the iterations

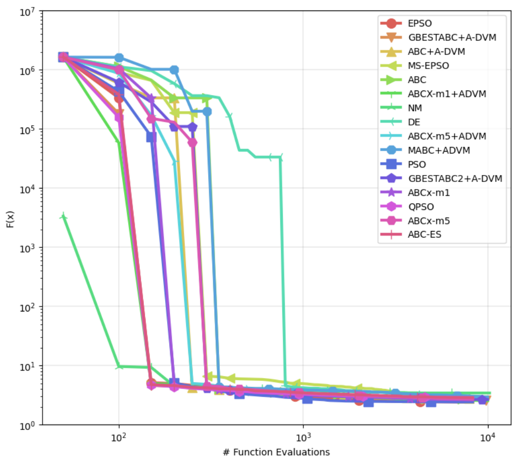


(a) MWTCS

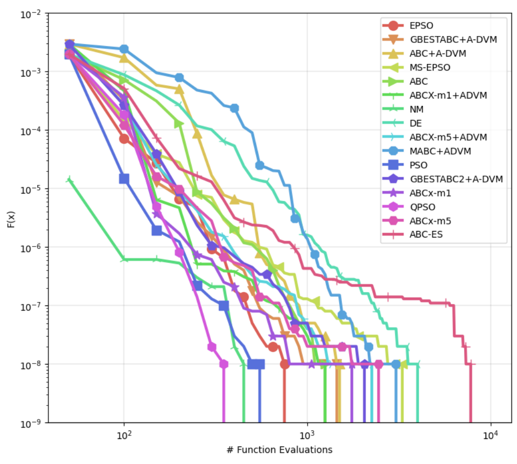


(b) WBD1

Figure 5.4: Plots of the objective function of the MWTCS and WBD1 along the iterations



(a) WBD2



(b) DTG

Figure 5.5: Plots of the objective function of the WBD2 and GTD along the iterations

Region	Method	Best	Mean	Worst	Std.Dev	Median
I	Sandgren[101]	8129.1036	-	-	-	-
	Kannan and Kramer[92]	7198.0428	-	-	-	-
	Coello[102]	6288.7445	6293.8432	6308.1497	7.4133	-
	Coello and Montes[103]	6059.9463	6177.2533	6469.3220	130.9297	-
	He and Wang[104]	6061.0777	6147.1332	6363.8041	86.4545	-
	Montes and Coello[105]	6059.7456	6850.0049	7332.8798	426.0000	-
	Kaveh and Talatahari[106]	6059.7258	6081.7812	6150.1289	67.2418	-
	Kaveh and Talatahari[107]	6059.0925	6075.2567	6135.3336	41.6825	-
	Garg[90]	6059.714	6447.7360	6495.3470	502.693	-
	Cagnina et al.[93]	6059.714335	6092.0498	-	12.1725	-
	Coelho[108]	6059.7208	6440.3786	7544.4925	448.4711	
	He et al. [109]	6059.7143	6289.92881	-	305.78	-
	Akay and Karaboga[12]	6059.714339	6245.308144	-	205	-
	Garg[90]	5885.3853363	5884.24637	5884.462541	0.50281	5884.58128
	c-NM	5915.96	6263.62	7580.9	325.158	6203.62
	GBESTABC+A-DVM	5838.33	5996.02	6478.85	130.14	5968.83
II	Dimopoulos[110]	5850.38306	-	-	-	-
	Mahdavi et al.[111]	5849.7617	-	-	-	-
	Garg[90]	5850.38306	5937.33790	5811.977127	264.54747	-
	Garg[90]	5804.4048008	5806.596206	5808.16968	1.028072	5806.7764199
	c-NM	5822.05	6155.07	7004.11	278.569	6074.56
	GBESTABC+A-DVM	5919.27	6113.5	6827.4	182.972	6050.65

Table 5.7: Results of the DPVI and II throughout the literature

Method	Best	Mean	Worst	Std Dev	Median
Kuang et al.[112]	2876.117623	-	-	-	-
Ray and Saini[113]	2732.9006	2741.5642	2757.8581	-	-
Akhtar et al.[114]	3008.08	3012.12	3028	-	-
Montes et al.[115]	3025.005	3088.7778	3078.5918	-	-
Ray and Liew[99]	2994.744241	3001.7582264	3009.964736	4.0091423	3001.758264
Montes et al.[116]	2996.356689	2996.367220	-	8.2×10^{-3}	-
Akay and Karaboga[12]	2996.348165	2996.3482	-	0	-
Zhang et al.[100]	2994.471066	2994.471066	2994.471066	3.58×10^{-12}	2994.471066
Garg[90]	3000.9810	3007.1997	-	4.9634	-
Garg[90]	2894.73832	2894.71248	2895.03219	4.96×10^{-4}	2894.97128
c-NM	3450.62	5493.69	14992.5	4056.27	3450.62
GBESTABC+A-DVM	2894.42	2894.39	2894.42	0.007253	2894.38

Table 5.8: Results of the SRD11 throughout the literature

Method	Best	Mean	Worst	Std.Dev	Median
Belegundu[98]	0.0128334	-	-	-	-
Coello [102]	0.01270478	0.01276920	0.01282208	3.9390×10^{-5}	0.01275576
Ray and Saini[113]	0.0130600	0.015526	0.018992	-	-
Coello and Montes[103]	0.0126810	0.0126810	0.012973	5.9000×10^{-5}	-
Ray and Liew[99]	0.01266924934	0.012922669	0.012709	5.92×10^{-4}	0.012922669
He et. al[109]	0.012922669	0.01270233	-	4.12439×10^{-5}	-
He and Wang[104]	0.0126747	0.012730	0.012924	5.1985×10^{-5}	-
Zhang et al.[100]	0.012665233	0.012669366	0.012738262	1.25×10^{-5}	-
Montes et al.[116]	0.0126747	0.012666	-	2.0×10^{-6}	-
Montes and Coello[105]	0.012698	0.013461	0.164850	9.6600×10^{-4}	-
Akay and Karaboga[12]	0.012665	0.0131	-	4.1×10^{-4}	-
Kaveh and Talatahari[107]	0.0126432	0.012720	0.012884	3.4888×10^{-5}	-
Coelho[108]	0.012665	0.013524	0.017759	0.001268	0.012957
Akay and Karaboga[12]	0.012665	0.012709	-	0.012813	-
Garg[90]	0.01266523278	0.01266523278	0.012668306480	8.6519×10^{-7}	0.01266555276
c-NM	0.0126665	0.0253572	0.236589	0.0459053	0.0126886
ABC+A-DVM	0.0129082	0.0137128	0.016592	0.00090146	0.0134881

Table 5.9: Results of the MWTCS throughout the literature

Region	Method	Best	Mean	Worst	Std.Dev	Median
I	Ragsdell and Phillips[117]	2.385937	-	-	-	-
	Rao[118]	2.3860	-	-	-	-
	Deb[119]	2.38119	-	-	-	
	Ray and Liew[99]	2.3854347	3.2551371	6.3996785	0.9590780	3.0025883
	Hwang and He[120]	2.25	2.26	2.28	-	-
	Mehta and Dasgupta[121]	2.381134	2.3811786	2.3812614	-	2.3811641
	Garg[90]	2.38099617	2.38108932	2.38146999	1.182265×10^{-4}	2.3786824
	c-NM	2.33394	2.66087	4.36388	0.427917	2.51524
	ABCX-m5+A-DVM	2.45812	2.63372	2.89238	0.130541	2.58517
II	Coello [102]	1.748309	1.771973	1.785835	0.011220	-
	Coello and Montes[103]	1.728226	1.792654	1.993408	0.07471	-
	Dimopoulos[110]	1.731186	-	-	-	-
	He and Wang[104]	1.728024	1.748831	1.782143	0.012926	-
	Montes et al.[116]	1.724852	1.725	-	0	-
	Montes and Coello[105]	1.737300	1.813290	1.994651	0.07050	-
	Akay and Karaboga[12]	1.724852	1.729752	-	0.21545	-
	Kaveh and Talatahari[106]	1.724918	1.727564	1.775961	0.0092	-
	Kaveh and Talatahari[107]	1.724849	1.8786560	1.759522	0.008254	-
	Mehta and Dasgupta[121]	1.724855	2.0574	1.72489	-	1.724861
	Akay and Karaboga[12]	1.724852	1.741913	-	0.031	-
	Garg[90]	1.69524738	1.6952473	1.6952473	1.978×10^{-8}	1.6952473
	c-NM	1.66504	3.4216	8.62822	1.62804	3.4927
	ABCX-m1+A-DVM	2.41082	2.53591	2,87406	0.110458	2.51429

Table 5.10: Results of the WBDI and II throughout the literature

Method	Best	Mean	Worst	Std Dev	Median
Gandomi et al.[88]	2.7009×10^{-12}	1.9841×10^{-9}	2.3576×10^{-9}	3.5546×10^{-9}	-
Garg[90]	2.7008571×10^{-9}	1.2149276×10^{-9}	3.2999231×10^{-9}	8.77×10^{-10}	$9.9215795 \times 10^{-10}$
c-NM	0	0	0	0	0
GBESTABC+A-DVM	0	0	0	0	0

Table 5.11: Results of the GTD throughout the literature

formulation in which specifically focuses the search around the feasible boundary of the problem. Naturally, they take precedence over the standard construction procedure.

Numerical experiments were conducted on eight instances of constrained engineering design problems in order to provide a suitable answer to the research question. The c-NM and A-DVM based ABC variants with the new rules are compared against several population based heuristics in order to assess its performance as a standalone method. The algorithms are firstly compared against a smaller pool of population heuristics as baseline, and then against a larger selection found in the literature. Results show that the augmented A-DVM was indeed able to reach competitive results in some instances, comparable to complex parallel methods. A similar thing can be said about the c-NM, albeit its performance was not as good compared to the A-DVM based algorithms. A deterioration of the robustness of the c-NM has been observed due to the sensitivity of the method to the choice of initial search point. Therefore, implementation of methods to choose initial points that are ”good enough” would yield much better results.

From the results, we can conclude that our objective has been achieved, that is, to come up with derivative-free techniques that rely more on deterministic procedures which are able to hold their ground against more complex algorithms. We can highlight two possible directions for the c-NM. First, to explore more on the property of the faces of the simplex rather than the vertices itself, so that it can better adapt itself to the landscape of the problem. Second, to capitulate that the algorithm needs a good starting point and employ the proposed approach as a subroutine to global search algorithms or even to the population based heuristics that were used in the experiment. After the integration, it would be fruitful to test the method to optimization problems with constraints that are so difficult which regular solvers lack the capability to solve. An example of such are problems from the family of mathematical programming with equilibrium constraints [122]. Lastly, for the A-DVM, a further understanding of the constraint rules would be a promising due to its competitive results. Furthermore, using a larger pool of engineering problem instances and compared against the state-of-the-art would prove to be ideal.

Chapter 6

Conclusion

The main point of this thesis was the proposal of several modifications based on deterministic methods to two derivative-free algorithms in order to improve their robustness to solve some families of optimization problems pertaining to the class of multimodal problems. The algorithms in question are the Artificial Bee Colony, a population heuristic from the family of swarm intelligence algorithms, and the Nelder-Mead algorithm, a direct search algorithm. Each modification resulted in novel algorithms that exploited the problem formulation by overcoming some of its deficiencies brought by employing randomization in their core structure. We restrict the numerical experiments used to validate the research question for each case to instances with relatively low dimensionality ($n < 100$) for two reasons. First, because very large scale are out of scope of this work, since one of our main focus is to try to keep the changes as simple as possible, it would be unfair to compare our techniques against the very complex algorithms for these families that are prominent in the literature. Second, simply due to the causality of the ”no-free lunch” theorem of Wolpert [64] in asserts the idea that the ideas presented in this work would be good at every kind of problem. We can affirm that we achieved our objective in Chapter 3, since the A-DVM based ABC algorithm was strictly better than the standard ABC for almost every case. Not only that, but the A-DVM approach has seen to be very robust to the hardest instances of problems, also faring well against other derivative-free techniques when restricting the budget to a very small number. On a last note, it was possible to observe that the A-DVM failed to obtain a competitive result for large scale cases, leaving room for future improvement in the deterministic selection process.

The same can be said for chapter 4, which dealt with an integration of a procedure that is commonplace in model-based algorithms to a direct search derivative-free algorithm. Although the use of polling steps to a direct search algorithm has resulted in some well-known algorithm such as the Multidirectional Direct Search (MADS), the inclusion of a polling step to reinforce the restarting mechanism of the solution set seemed a natural idea since it is common knowledge that the Nelder-Mead is extremely reliant on the starting point to construct the initial solution set.

Lastly, Chapter 5 held mixed results. On one hand, the augmented Lagrangian together with the additional rules for the deterministic selection of the A-DVM provided a substantial improvement to the ABC algorithm. On the other hand, the new mechanism proposed to the Nelder-Mead tied with the Lagrangian penalty method was not able to provide an improvement that was up to the expectations. Surely, we can conclude that for the inclusion of the A-DVM is a promising direction towards obtaining robust results, especially compared against the state-of-the-art. Nevertheless, the c-NM cannot be seen as a complete failure, since the results strongly suggest that it would perform much better if it would be implemented as a subroutine of another global optimization algorithm.

Acknowledgements

Foremost, I would like to express my sincerest gratitude to my advisors, Professor Takahito Kuno and Professor Yoshio Sano for the continuous support of my study and research, for their patience, motivation, enthusiasm, and immense knowledge. And of course standing by my side for such a long time. Besides my advisors, I also would like to extend my thanks to two professors in particular: Professor Claus Aranha for being part of the thesis committee once again and keeping it very real, also Professor Marcos Neves being a fatherly figure during all the years that I spent here.

I cannot forget about my partner in crime, Bernardo Gatto who was my co-author to the great majority of my scientific contributions. Our partnership is a fruitful one and I am sure that in the near future we will be able to accomplish even further.

Of course, my utmost and sincerest thanks to my beloved mother, Franci Florenzano. Once again, I could not have done this without your help. You encouraged me, helped me in way that you cannot imagine. I wish I could repay at least a fraction of what you have done for me so far. I cannot forget to once again pay homage to my deceased father, Marco Antônio Mollinetti. I am once again reaching for my dreams, hope you are seeing me and proud from up there! I extend my heartfelt thanks to all my relatives too, although I was a bit far from all these years, I definitely received all the positive energies.

I also want to extend my heartfelt thanks to my cat Kyo, I miss him every day and I sure wish I could pet him.

I could not forget to thank my colleagues in the Computational Intelligence Laboratory back in Brazil. My former mentors, Otávio Noura and Rodrigo Lisbôa, as well as my colleagues, Adilson Neto and Mário Tasso. I would like to give a special thanks to all my friends in Brazil who were fundamental to help me carry on at my hardest. A heartfelt thanks to all my colleagues/friends who I have the pleasure to work at Plimes, and all of the people that I befriended here in Japan. I will not write the names because otherwise it would be a very long list, but know that I thank you, you, you and you!

A heartfelt thanks to The Ministry of Education, Culture, Sports, Science and Technology, (MEXT) for offering me the possibility of embarking in this journey to pursue a higher education.

Lastly, I thank God, and the whole Pantheon of the other gods out there. In times of distress I reached out to every single one of them.

Bibliography

[1] Stephen G Nash. *Linear and nonlinear programming*. McGraw-Hill Science, Engineering & Mathematics, 1996.

[2] Andrew R Conn, Katya Scheinberg, and Luis N Vicente. *Introduction to derivative-free optimization*. SIAM, 2009.

[3] D. Karaboga. An idea based on honey bee swarm for numerical optimization. Technical report, Erciyes University, 2005.

[4] Bahriye Akay and Dervis Karaboga. A survey on the applications of artificial bee colony in signal, image, and video processing. *Signal, Image and Video Processing*, Vol. 9, No. 4, pp. 967–990, 2015.

[5] Dervis Karaboga, Beyza Gorkemli, Celal Ozturk, and Nurhan Karaboga. A comprehensive survey: artificial bee colony (abc) algorithm and applications. *Artificial Intelligence Review*, Vol. 42, No. 1, pp. 21–57, 2014.

[6] Doğan Aydın, Gürcan Yavuz, and Thomas Stützle. Abc-x: a generalized, automatically configurable artificial bee colony framework. *Swarm Intelligence*, Vol. 11, No. 1, pp. 1–38, 2017.

[7] Bilal Alatas. Chaotic bee colony algorithms for global numerical optimization. *Expert Systems with Applications*, Vol. 37, No. 8, pp. 5682–5687, 2010.

[8] Bahriye Akay and Dervis Karaboga. A modified artificial bee colony algorithm for real-parameter optimization. *Information sciences*, Vol. 192, pp. 120–142, 2012.

[9] Weifeng Gao and Sanyang Liu. Improved artificial bee colony algorithm for global optimization. *Information Processing Letters*, Vol. 111, No. 17, pp. 871–882, 2011.

[10] Konrad Diwold, Andrej Aderhold, Alexander Scheidler, and Martin Middendorf. Performance evaluation of artificial bee colony optimization and new selection schemes. *Memetic Computing*, Vol. 3, No. 3, p. 149, 2011.

[11] Wan-Li Xiang and Mei-Qing An. An efficient and robust artificial bee colony algorithm for numerical optimization. *Computers & Operations Research*, Vol. 40, No. 5, pp. 1256–1265, 2013.

[12] Bahriye Basturk Akay and Dervis Karaboga. Artificial bee colony algorithm variants on constrained optimization. *An International Journal of Optimization and Control: Theories & Applications (IJOCTA)*, Vol. 7, No. 1, pp. 98–111, 2017.

[13] Sangeeta Sharma and Pawan Bhambu. Artificial bee colony algorithm: a survey. *International Journal of Computer Applications*, Vol. 975, p. 8887, 2016.

[14] John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, Vol. 7, No. 4, pp. 308–313, 1965.

[15] WGRFR Spendley, George R Hext, and Francis R Himsworth. Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics*, Vol. 4, No. 4, pp. 441–461, 1962.

[16] Margaret H Wright, et al. Nelder, mead, and the other simplex method. *Documenta Mathematica*, Vol. 7, pp. 271–276, 2010.

[17] Robert Michael Lewis, Virginia Torczon, and Michael W Trosset. Direct search methods: then and now. *Journal of computational and Applied Mathematics*, Vol. 124, No. 1-2, pp. 191–207, 2000.

[18] Luis Miguel Rios and Nikolaos V Sahinidis. Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, Vol. 56, No. 3, pp. 1247–1293, 2013.

[19] Jeffrey C Lagarias, James A Reeds, Margaret H Wright, and Paul E Wright. Convergence properties of the nelder–mead simplex method in low dimensions. *SIAM Journal on optimization*, Vol. 9, No. 1, pp. 112–147, 1998.

[20] Margaret H Wright. Direct search methods: Once scorned, now respectable. *Pitman Research Notes in Mathematics Series*, pp. 191–208, 1996.

[21] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[22] Ken IM McKinnon. Convergence of the nelder–mead simplex method to a nonstationary point. *SIAM Journal on Optimization*, Vol. 9, No. 1, pp. 148–158, 1998.

[23] JM Parkinson and D Hutchinson. An investigation into the efficiency of variants on the simplex method. *Numerical Methods for Nonlinear Optimization*, pp. 115–135, 1972.

[24] Shu-Kai S Fan and Erwie Zahara. A hybrid simplex search and particle swarm optimization for unconstrained optimization. *European Journal of Operational Research*, Vol. 181, No. 2, pp. 527–548, 2007.

[25] Peter C Wang and Terry E Shoup. Parameter sensitivity study of the nelder–mead simplex method. *Advances in Engineering Software*, Vol. 42, No. 7, pp. 529–533, 2011.

[26] Fuchang Gao and Lixing Han. Implementing the nelder-mead simplex algorithm with adaptive parameters. *Computational Optimization and Applications*, Vol. 51, No. 1, pp. 259–277, 2012.

[27] Saša Singer and Sanja Singer. Efficient implementation of the nelder–mead search algorithm. *Applied Numerical Analysis & Computational Mathematics*, Vol. 1, No. 2, pp. 524–534, 2004.

[28] Larry Nazareth and Paul Tseng. Gilding the lily: a variant of the nelder-mead algorithm based on golden-section search. *Computational Optimization and Applications*, Vol. 22, No. 1, pp. 133–144, 2002.

[29] Marco A Luersen and Rodolphe Le Riche. Globalized nelder–mead method for engineering optimization. *Computers & structures*, Vol. 82, No. 23, pp. 2251–2260, 2004.

[30] Virginia Torczon. On the convergence of pattern search algorithms. *SIAM Journal on optimization*, Vol. 7, No. 1, pp. 1–25, 1997.

[31] Sanja Singer and Saša Singer. Complexity analysis of nelder-mead search iterations. In *Proceedings of the 1. Conference on Applied Mathematics and Computation*, pp. 185–196, 1999.

[32] Stephen Smith. The simplex method and evolutionary algorithms. In *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360)*, pp. 799–804. IEEE, 1998.

[33] Paul Tseng. Fortified-descent simplicial search method: A general approach. *SIAM Journal on Optimization*, Vol. 10, No. 1, pp. 269–288, 1999.

[34] Charles Audet and Warren Hare. *Derivative-free and blackbox optimization*. Springer, 2017.

[35] Jeffrey C Lagarias, Bjorn Poonen, and Margaret H Wright. Convergence of the restricted nelder–mead algorithm in two dimensions. *SIAM Journal on Optimization*, Vol. 22, No. 2, pp. 501–532, 2012.

[36] Carl Tim Kelley. Detection and remediation of stagnation in the nelder–mead algorithm using a sufficient decrease condition. *SIAM journal on optimization*, Vol. 10, No. 1, pp. 43–55, 1999.

[37] Christopher John Price, Ian D Coope, and David Byatt. A convergent variant of the nelder–mead algorithm. *Journal of optimization theory and applications*, Vol. 113, No. 1, pp. 5–19, 2002.

[38] Charles Audet and John E Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on optimization*, Vol. 17, No. 1, pp. 188–217, 2006.

[39] CP Stephens and W Baritompa. Global optimization requires global information. *Journal of Optimization Theory and Applications*, Vol. 96, No. 3, pp. 575–588, 1998.

[40] Simon Wessing. Proper initialization is crucial for the nelder–mead simplex search. *Optimization Letters*, Vol. 13, No. 4, pp. 847–856, 2019.

[41] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Veerlag, 2 edition, 2006.

[42] Gabriel Jarry-Bolduc, Patrick Nadeau, and Shambhavi Singh. Uniform simplex of an arbitrary orientation. *Optimization Letters*, Vol. 14, No. 6, pp. 1407–1417, 2020.

[43] John E Dennis Jr and Daniel J Woods. Optimization on microcomputers. the nelder-mead simplex algorithm. Technical report, 1985.

[44] Raphael T Haftka and Zafer Gürdal. *Elements of structural optimization*, Vol. 11. Springer Science & Business Media, 2012.

[45] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, Vol. 17, pp. 261–272, 2020.

[46] MA El-Gebeily* and YA Fiagbedzi. On certain properties of the regular n-simplex. *International Journal of Mathematical Education in Science and Technology*, Vol. 35, No. 4, pp. 617–629, 2004.

[47] Marco Antonio Florenzano Mollinetti, Mario Tasso Ribeiro Serra Neto, and Takahito Kuno. Deterministic parameter selection of artificial bee colony based on diagonalization. In *International Conference on Hybrid Intelligent Systems*, 2018.

[48] Ronald W Morrison. *Designing evolutionary algorithms for dynamic environments*. Springer Science & Business Media, 2013.

[49] Marco Locatelli and Fabio Schoen. *Global optimization: theory, algorithms, and applications*, Vol. 15. Siam, 2013.

[50] Eric W Weisstein. *CRC concise encyclopedia of mathematics*. Chapman and Hall/CRC, 2002.

[51] Elias Zakon. *Mathematical analysis*. The Trillia Group, 2004.

[52] Brian Mc Ginley, John Maher, Colm O'Riordan, and Fearghal Morgan. Maintaining healthy population diversity using adaptive crossover, mutation, and selection. *IEEE Transactions on Evolutionary Computation*, Vol. 15, No. 5, pp. 692–714, 2011.

[53] Rasmus K Ursem. Diversity-guided evolutionary algorithms. In *International Conference on Parallel Problem Solving from Nature*, pp. 462–471. Springer, 2002.

[54] Thomas Bäck and Frank Hoffmeister. Extended selection mechanisms in genetic algorithms. pp. 92–99. Morgan Kaufmann, 1991.

[55] Andrea Gavana. Global optimization benchmarks and ampgo. *Accessed Apr*, 2019.

[56] Momin Jamil and Xin-She Yang. A literature survey of benchmark functions for global optimization problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, Vol. 4, No. 2, pp. 150–194, 2013.

[57] Marcin Molga and Czesław Smutnicki. Test functions for optimization needs. *Test functions for optimization needs*, p. 101, 2005.

[58] Ali R. Al-Roomi. Unconstrained Single-Objective Benchmark Functions Repository, 2015.

[59] Dervis Karaboga. An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.

[60] Weifeng Gao, Sanyang Liu, and Lingling Huang. A global best artificial bee colony algorithm for global optimization. *Journal of Computational and Applied Mathematics*, Vol. 236, No. 11, pp. 2741–2753, 2012.

[61] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of the IEEE Int. Conf. on Neural Networks*, pp. 1942–1948. IEEE, IEEE Press, 1995.

[62] Vladimiro Miranda and Nuno Fonseca. Epso-evolutionary particle swarm optimization, a new algorithm with applications in power systems. In *Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES*, Vol. 2, pp. 745–750. IEEE, 2002.

[63] Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, Vol. 11, No. 4, pp. 341–359, 1997.

[64] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, Vol. 1, No. 1, pp. 67–82, 1997.

[65] David J Wales and Jonathan PK Doye. Global optimization by basin-hopping and the lowest energy structures of lennard-jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A*, Vol. 101, No. 28, pp. 5111–5116, 1997.

[66] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, Vol. 11, No. 1, pp. 1–18, 2003.

[67] Wyn L. Price. A controlled random search procedure for global optimisation. *The Computer Journal*, Vol. 20, No. 4, pp. 367–370, 1977.

[68] JM Gablonsky. An implementation of the direct algorithm. Technical report, North Carolina State University. Center for Research in Scientific Computation, 1998.

[69] Xin-She Yang and Xingshi He. Firefly algorithm: recent advances and applications. *International journal of swarm intelligence*, Vol. 1, No. 1, pp. 36–50, 2013.

[70] J. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[71] AHG Rinnooy Kan and Gerrit T Timmer. Stochastic global optimization methods part i: Clustering methods. *Mathematical programming*, Vol. 39, No. 1, pp. 27–56, 1987.

[72] J. Kennedy and R. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, 2 edition, 2001.

[73] Qingyun Duan, Soroosh Sorooshian, and Vijai Gupta. Effective and efficient global optimization for conceptual rainfall-runoff models. *Water resources research*, Vol. 28, No. 4, pp. 1015–1031, 1992.

[74] Peter JM Van Laarhoven and Emile HL Aarts. Simulated annealing. In *Simulated annealing: Theory and applications*, pp. 7–15. Springer, 1987.

[75] Warren Hare and Gabriel Jarry-Bolduc. Calculus identities for generalized simplex gradients: Rules and applications. *SIAM Journal on Optimization*, Vol. 30, No. 1, pp. 853–884, 2020.

[76] Ana Luısa Custódio and Luís Nunes Vicente. Using sampling and simplex derivatives in pattern search methods. *SIAM Journal on Optimization*, Vol. 18, No. 2, pp. 537–555, 2007.

[77] Lixing Han and Michael Neumann. Effect of dimensionality on the nelder–mead simplex method. *Optimization Methods and Software*, Vol. 21, No. 1, pp. 1–16, 2006.

[78] Jorge J Moré, Burton S Garbow, and Kenneth E Hillstrom. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software (TOMS)*, Vol. 7, No. 1, pp. 17–41, 1981.

[79] Rommel G Regis. On the properties of positive spanning sets and positive bases. *Optimization and Engineering*, Vol. 17, No. 1, pp. 229–262, 2016.

[80] Ian D Coope and Christopher John Price. Positive bases in numerical optimization. *Computational Optimization and Applications*, Vol. 21, No. 2, pp. 169–175, 2002.

[81] Charles Audet. A short proof on the cardinality of maximal positive bases. *Optimization Letters*, Vol. 5, No. 1, pp. 191–194, 2011.

[82] Mário Serra Neto, Marco Mollinetti, Vladimiro Miranda, and Leonel Carvalho. Maximum search limitations: Boosting evolutionary particle swarm optimization exploration. In *EPIA Conference on Artificial Intelligence*, pp. 712–723. Springer, 2019.

[83] Jun Sun, Wenbo Xu, and Bin Feng. A global search strategy of quantum-behaved particle swarm optimization. In *IEEE Conference on Cybernetics and Intelligent Systems, 2004.*, Vol. 1, pp. 111–116. IEEE, 2004.

[84] Efren Mezura-Montes. Constraint-handling in evolutionary optimization, 2009.

[85] Ernesto G Birgin and José Mario Martínez. *Practical augmented Lagrangian methods for constrained optimization*. SIAM, 2014.

[86] D. M. Himmelblau. *Applied Nonlinear Programming*. McGraw-Hill, 1972.

[87] Marco Antônio Florenzano Mollinetti, Daniel Leal Souza, Rodrigo Lisbôa Pereira, Edson Koiti Kudo Yasojima, and Otávio Noura Teixeira. Abc+ es: Combining artificial bee colony algorithm and evolution strategies on engineering design problems and benchmark functions. In *Hybrid Intelligent Systems*, pp. 53–66. Springer, 2016.

[88] A. Gandomi, X. S. Yang, and A. Alavi. Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers*, Vol. 29, No. 1, pp. 17–35, April 2013.

[89] M. Zhang, W. Luo, and X. Wang. Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences*, Vol. 178, No. 15, pp. 3043–3074, August 2008.

[90] H. Garg. A hybrid gsa-ga algorithm for constrained optimization problems. *Information Sciences*, Vol. 478, pp. 499–523, April 2019.

[91] E. Sandgren. Nonlinear integer and discrete programming in mechanical design optimization. *Proceedings of the ASME Design Technology Conference*, Vol. 112, No. 2, pp. 223–229, June 1990.

[92] B. Kannan and S. Krammer. An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design*, Vol. 116(2), pp. 405–411, 1994.

[93] L. Cagnina, S. Esquivel, and C. Coello Coello. Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Informatica,*, Vol. 32(3), pp. 319–326, 2008.

[94] H. Garg. Solving structural engineering design optimization problems using an artificial bee colony algorithm. *Journal of Industrial and Management Optimization*, Vol. 10, No. 3, pp. 777–794, July 2014.

[95] H. Garg. *Handbook of Research on Artificial Intelligence Techniques and Algorithms*, chapter A Hybrid GA - GSA Algorithm for Optimizing the Performance of an Industrial System by Utilizing Uncertain Data, pp. 620–654. IGI Global, 2015.

[96] J. Golinski. An adaptive optimization system applied to machine synthesis. *Mech. Mach. Synthesis*, Vol. 8(4), pp. 419–436, 1973.

[97] J. Arora. *Introduction to Optimum Design*. Academic Press, 2 edition, 2004.

[98] A. Belegundu and J. Arora. A study of mathematical programming methods for structural optimization. part i: Theory. *International Journal for Numerical Methods in Engineering*, Vol. 21(9), pp. 1583–1599, 1985.

[99] Tapabrata Ray and Kim-Meow Liew. Society and civilization: An optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation*, Vol. 7, No. 4, pp. 386–396, 2003.

[100] Min Zhang, Wenjian Luo, and Xufa Wang. Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences*, Vol. 178, No. 15, pp. 3043–3074, 2008.

[101] J. Sanders and E. Kandrot. *CUDA By Example: An Introduction to General-Purpose GPU Programming*. Addison-Wesley Professional, 1 edition, 2010.

[102] C. Coello Coello. Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, Vol. 41(2), pp. 113–127, 2000.

[103] C. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, Vol. 191(11-12), pp. 1245–1287, 2002.

[104] Qie He and Ling Wang. An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering applications of artificial intelligence*, Vol. 20, No. 1, pp. 89–99, 2007.

[105] E. M. Montes and C. A. C. Coello. An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *Internation Journal of General Systems*, Vol. 37, No. 4, pp. 443–473, July 2008.

[106] A Kaveh and S Talatahari. Engineering optimization with hybrid particle swarm and ant colony optimization. 2009.

[107] A Kaveh and S Talatahari. An improved ant colony optimization for constrained engineering design problems. *Engineering Computations*, 2010.

[108] Leandro dos Santos Coelho. Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. *Expert Systems with Applications*, Vol. 37, No. 2, pp. 1676–1683, 2010.

[109] S He, E Prempain, and QH Wu. An improved particle swarm optimizer for mechanical design optimization problems. *Engineering optimization*, Vol. 36, No. 5, pp. 585–605, 2004.

[110] George G Dimopoulos. Mixed-variable engineering optimization based on evolutionary and social metaphors. *Computer methods in applied mechanics and engineering*, Vol. 196, No. 4-6, pp. 803–817, 2007.

[111] Mehrdad Mahdavi, Mohammad Fesanghary, and E Damangir. An improved harmony search algorithm for solving optimization problems. *Applied mathematics and computation*, Vol. 188, No. 2, pp. 1567–1579, 2007.

[112] K'uang J Ku, Singiresu S Rao, and Li Chen. Taguchi-aided search method for design optimization of engineering systems. *Engineering Optimization*, Vol. 30, No. 1, pp. 1–23, 1998.

[113] Tapabrata Ray and Pankaj Saini. Engineering design optimization using a swarm with an intelligent information sharing among individuals. *Engineering Optimization*, Vol. 33, No. 6, pp. 735–748, 2001.

[114] Shamim Akhtar, Kang Tai, and Tapabrata Ray. A socio-behavioural simulation model for engineering design optimization. *Engineering Optimization*, Vol. 34, No. 4, pp. 341–354, 2002.

[115] Efrén Mezura-Montes, CA Coello Coello, and Ricardo Landa-Becerra. Engineering optimization using simple evolutionary algorithm. In *Proceedings. 15th IEEE international conference on tools with artificial intelligence*, pp. 149–156. IEEE, 2003.

[116] Efrén Mezura-Montes, CA Coello Coello, Jesús Velázquez-Reyes, and Lucıa Muñoz-Dávila. Multiple trial vectors in differential evolution for engineering design. *Engineering Optimization*, Vol. 39, No. 5, pp. 567–589, 2007.

[117] KM Ragsdell and DT Phillips. Optimal design of a class of welded structures using geometric programming. 1976.

[118] S. S. Rao. *Engineering optimization: Theory and Practice*. No. 3rd. John Wiley and Sons, 1996.

[119] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms. *Computer methods in applied mechanics and engineering*, Vol. 186, No. 2-4, pp. 311–338, 2000.

[120] Shun-Fa Hwang and Rong-Song He. A hybrid real-parameter genetic algorithm for function optimization. *Advanced Engineering Informatics*, Vol. 20, No. 1, pp. 7–21, 2006.

[121] V. K. Mehta and B. Dasgupta. A constrained optimization algorithm based on the simplex search method. *Engineering Optimization*, Vol. 44, No. 5, pp. 537–550, September 2012.

[122] Zhi-Quan Luo, Jong-Shi Pang, and Daniel Ralph. *Mathematical programs with equilibrium constraints*. Cambridge University Press, 1996.

List of Publications

Journal Articles

1. Mollinetti, Marco Antonio Florenzano and Gatto, Bernardo Bentes and Neto, Mário Tasso Ribeiro Serra and Kuno, Takahito: **A-DVM: A Self-Adaptive Variable Matrix Decision Variable Selection Scheme for Multimodal Problems**. – *Entropy*, 22(9), (2020)

2. Pereira, Rodrigo Lisbôa and Souza, Daniel Leal and Mollinetti, Marco Antônio Florenzano and Neto, Mário TR Serra and Yasojima, Edson Koiti Kudo and Teixeira, Otávio Noura and De Oliveira, Roberto Celio Limão: **Game Theory and Social Interaction for Selection and Crossover Pressure Control in Genetic Algorithms: An Empirical Analysis to Real-Valued Constrained Optimization**. – *IEEE Access*, 8(1), (2020), pp.144839–144865

3. Mollinetti, Marco Antonio Florenzano and Gatto, Bernardo Bentes and Kuno, Takahito: **A Modified Nelder-Mead with Polling for Constrained Optimization**. – *Global Optimization Letters*, To be published, (2021)

Conference Papers

1. Mollinetti, Marco Antonio Florenzano and Neto, Mario Tasso Ribeiro Serra and Kuno, Takahito: **Deterministic parameter selection of artificial bee colony based on diagonalization**. – *International Conference on Hybrid Intelligent Systems*, Porto, Portugal, December 2018.

2. Mollinetti, Marco and Gatto, Bernardo and Neto, Mario and Kuno, Takahito: **Selecting Decision Variables for Artificial Bee Colony using a Self-adaptive Variable Matrix**. – *XVI Brazilian Computational Intelligence and Artificial Intelligence*, Salvador, Brazil, October 2019.

3. Gatto, Bernardo B and Molinetti, Marco AF and dos Santos, Eulanda M and Fukui, Kazuhiro: **Tensor Fukunaga-Koontz Transform for Hierarchical Clustering**. – *8th Brazilian Conference on Intelligent Systems (BRACIS)*, Salvador, Brazil, October 2019.