J COMP SCI APPL MATH
Journal of Computer Science and Applied Mathematics
Vol. 5, no.2, (2023), 53–73
ISSN: 1857-9582 https://doi.org/10.37418/jcsam.5.2.2

PERFORMANCE STUDY OF MULTIOBJECTIVE OPTIMIZER METHOD BASED ON GREY WOLF ATTACK TECHNICS

Wendinda Bamogo, Kounhinir Some[1], and Guy Aymard Degla

Abstract. This paper proposes a performance study for the Multiobjective Optimizer based on the Grey Wolf Attack technics (MOGWAT). It is a method of solving multiobjective optimization problems. The method consists of the resolution of an unconstrained single objective optimization problem, which is derived from the aggregation of objective functions by the ϵ-constraint approach and the penalization of constraints by a Lagrangian function. Then, Pareto-optimal solutions are obtained using the stochastic method based on the Grey Wolf Optimizer. To evaluate the method, three theorems have been formulated to demonstrate the convergence of the proposed algorithm and the optimality of the obtained solutions. Six test problems from the literature have been successfully dealt with, and the obtained results have been compared to two other methods. We have evaluated two performance parameters, including the generational distance for the approximation error and the spread for the coverage of the Pareto front. Based on these numerical findings, it can be concluded that MOGWAT outperforms two other methods.

1. Introduction

For the solution of many real-world problems, a multiobjective optimization model is used. That is a mathematical programming formula where multiple objective functions are considered at the same time to minimize or/and maximize.

[1] *corresponding author*
Key words and phrases. Multiobjective optimization, Grey Wolf Optimizer, Pareto optimality.
Submitted: 08.09.2023; *Accepted:* 23.09.2023; *Published:* 30.09.2023.

In many cases, these objective functions are submitted to constraints. These models do not have a unique solution, but many solutions. In this area, these kinds of solutions are called Pareto optimal solutions. We have two types of methods for solving these problems: exact methods [22,23,25] and metaheuristic methods [1,6,8,12]. Exact methods have difficulties in solving problems in which we have many data, such as variables or objective functions or constraint functions. Metaheuristic methods are the most common, but they usually lead to finding a good approximation of the Pareto optimal solutions set. The goodness of solutions is evaluated on three measures: convergence of the obtained solutions to the true Pareto front, distribution of obtained solutions on the true Pareto front, and computational time of obtained solutions. It is almost impossible to find a method that fulfills all these evaluation measures. In practice, it is difficult to find one of the existing methods from the literature that can be applied to all kinds of multiobjective optimization problems.

Metaheuristic methods, in general, are methods that are inspired by the natural phenomena of life [1,2,4] for the resolution of complex problems. Initial, they were designed for a single objective problem, and thereafter they were adapted for multiple objective optimization problems resolution. Among these kinds of methods, we cite genetic algorithm (GA) [11] and grey wolf optimizer (GWO) [10]. Numerous studies have proposed approaches that combine these two algorithms, and HmGWOGA [9] is one of them. This has been built to optimize single-positive objective functions without constraints. MOGWAT [8] is its extension to resolve multiple objectives cases. It works by transforming the optimization of multiple objective functions with constraints into the optimization of a single objective function without constraints by using the ϵ-constraint approach [1] and the Lagrangian penalty function [23]. It is a method from the literature, but no theoretical convergence study has been conducted to prove the convergence of its algorithm.

Therefore, in this paper, we propose a performance study that considers both the numerical and theoretical performance. This work establishes the convergence of the MOGWAT method through studies of the consistency and stability of its algorithm. Then, this work illustrates the numerical capabilities of the MOGWAT method by comparing it to two other methods, namely MSSA [13] and

MOGOA [12]. This comparison is done on six test problems from the literature and is focused on convergence of obtained solutions toward the true Pareto front and the distribution of obtained solutions on the Pareto front. Based on these results, we have concluded that the MOGWAT method is efficient and effective at finding Pareto optimal solutions to multiobjective optimization problems.

In order to facilitate the comprehension of this document, we shall arrange it in the following manner. Section 2 will present the preliminary. Section 3 will be dedicated to highlighting the main results of this work; Section 5 will be dedicated to conclusion.

2. Preliminary

2.1. Multiobjective optimization concepts.

Let $f = (f_1, f_2, \dots, f_p)$ and $g = (g_1, g_2, \dots, g_m)$ be numerical vector functions with p and m some finite integer numbers. Multiobjective optimization problems, especially the cases of minimization, are formulated mathematically by:

$$\text{(MOP)} \qquad \begin{cases} \min\left(f_1(x), f_2(x), \dots, f_p(x)\right), p \geq 2 \\ g_j(x) \leq 0, j \in \overline{1:m} \\ x \in \mathbb{R}^n \end{cases}$$

where $f_i : \mathbb{R}^n \to \mathbb{R},\ i = 1, 2, \dots, p$, is an objective function and $g_j : \mathbb{R}^n \to \mathbb{R},\ j = 1, 2, \dots, m$ is a constraint function. In the follow, we will put $\chi = \{x \in \mathbb{R}^n : g_i \leq 0\}$ and $\mathcal{Y} = f(\chi)$ respectively the decision space and the objective space.

Definition 2.1. *A point $x^* \in \chi$ is called Pareto optimal solution of problem (MOP) if there is no other point $x \in \chi$ such that $f(x) \leq f(x^*)$ and $f(x) \neq f(x^*)$.*

In this case, $f(x^*) = \left(f_1(x^*), f_2(x^*), \dots, f_p(x^*)\right)$ is said a non-dominated point. That allows us to set $\mathcal{P}_s = \{x^* \in \chi : f(x^*) \text{ is non-dominated}\}$ as a Pareto optimal solutions set and $\mathcal{P}_f = \{(x, f(x)) : x \in \mathcal{P}_s\}$ Pareto front.

Definition 2.2. *A point $x^* \in \chi$ is called weakly Pareto optimal solution of problem (MOP) if there is no $x \in \chi$ for which $f(x) < f(x^*)$.*

By noting weakly Pareto optimal solutions $\overline{\mathcal{P}}_s$, we have $\mathcal{P}_s \subseteq \overline{\mathcal{P}}_s$.

To obtain these kinds of solutions, many methods try to transform multiple objective functions into a single objective function by using some aggregation function. The well-known and well-used are weighted sum, weighted Tchebychev distance, augmented weighted Tchebychev distance, ϵ-constraint approach [3,24]. In the MOGWAT method, it is the ϵ-constraint approach which was chosen to build the algorithm. This aggregation function has many advantages, such as using a few parameters to fix in advance, and preserving Pareto optimal solutions during the conversion from multiple objective functions to single objective function when we have any kind of problems.

The methods, which deal directly with the optimization of a single objective function under constraints, sometimes have difficulties obtaining optimal solutions. Therefore, in many methods, a step consisting of transforming the problem into an unconstrained is made before. In this framework, we will use the Lagrangian penalty function [22, 23, 25] to transform the problem into an unconstrained a single-objective optimization problem. This last form of the problem requires the use of a single-objective optimizer in order to get the optimal solution. At this step, the Grey wolf optimizer will be used.

2.2. **Grey Wolf Optimizer.**

The Grey Wolf Optimizer, as noted by GWO, is an algorithm inspired by the leadership and hunting process of grey wolves. They are predators that live in groups [18], in which we can identify four subgroups: the grey wolf α, the grey wolf β, the grey wolf δ, and grey wolves ω. This organization of the grey wolf hunt is represented by a triangle, as shown in the Figure 2.2 below.

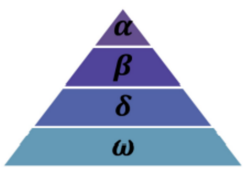

FIGURE 1. Hierarchy of grey wolf [10]

In practice, the hunting is done in group and is led by the grey wolves α, β, and δ, but each grey wolf is considered as a search agent. The first three subgroups are the best search agents in the group. The other grey wolves, ω, orientate their search position according to the positions of three subgroups of leaders. According to Muro et al. [18], the main steps of hunting are the following:

- ◇ track, pursue and approach the prey;
- ◇ encircle and harass the prey until it stops moving;
- ◇ Attack the prey.

These different steps can be given in the following Figure 2.2. That is presented in the works of C. Muro et al. [18].

FIGURE 2. Grey wolves Hunting steps

Figure 2.2 is subdivided in five parts: part A is chasing, approaching, and tracking; part B, C, and D are pursuiting, harassing, and encircling; and part E is stationary situation and attacking.

The following equations are used to present mathematical models of social hierarchy, pursuit, encirclement, and prey attack [10, 18].

$$\begin{cases} \vec{D}(t) = |\vec{C}.\vec{X}_p(t) - \vec{X}(t)| \\ \vec{X}(t+1) = \vec{X}_p(t) - \vec{A}.\vec{D}(t) \end{cases}, \tag{2.1}$$

where t is the current iteration number, $\vec{A} = 2\vec{a}\vec{r}_1 - \vec{a}$, $\vec{C} = 2\vec{r}_2$ are some random vectors; the vector $\vec{a} = 2\Big(1 - \dfrac{t^d}{T^d}\Big)$ linearly decreasing from 2 to O during the iterations; $\vec{r}_1,\ \vec{r}_2$ are random vectors taken in $[0,1]$; $\vec{X}_p$ is the position vector of the prey; $\vec{X}$ is the vector given the position of a grey wolf; and T correspond to the maximal number of the iterations.

During the attack, the position of each grey wolf in the group is changed based on the position of the leaders α, β and δ. Following are the equations that provide the novel positions of these leaders.

$$\begin{cases} D^i_\alpha(t) = |C_\alpha G_\alpha(t) - X_i(t)|, V^i_\alpha(t) = G_\alpha(t) - A_\alpha D^i_\alpha(t) \\ D^i_\beta(t) = |C_\beta G_\beta(t) - X_i(t)|, V^i_\beta(t) = G_\beta(t) - A_\beta D^i_\beta(t) \\ D^i_\delta(t) = |C_\delta G_\delta(t) - X_i(t)|, V^i_\delta(t) = G_\delta(t) - A_\delta D^i_\delta(t) \\ X_i(t+1) = \lambda_\alpha V^i_\alpha(t) + \lambda_\beta V^i_\beta(t) + \lambda_\delta V^i_\delta(t), i = 1, \dots, N \end{cases} , \tag{2.2}$$

where N is the number of search agents and λ_α, λ_β, λ_δ are the weights such as $\lambda_\alpha + \lambda_\beta + \lambda_\delta = 1$.

3. Main results

Let us recall the formulation of multiobjective optimization problem:

$$\text{(MOP)} \qquad \begin{cases} \min\left(f_1(x), f_2(x), \dots, f_p(x)\right), p \geq 2 \\ g_j(x) \leq 0, \ j \in \overline{1:m} \\ x \in \mathbb{R}^n \end{cases} ,$$

where $p, m, n \in \mathbb{N}$.

3.1. MOGWAT method description.

MOGWAT is an iterative and stochastic method that uses an initial population of solutions to provide the best ones for a given multiobjective optimization problem. We can identify two main steps: conversion of the initial problem to an unconstrained single-objective problem and minimize of a function.

To get an unconstrained single-objective function, that requires the using of an aggregation function followed by the using of the penalty function. In the MOGWAT method, the aggregation is realized with the use of ϵ-constraint approach [8] and the penalization through a Lagrangian function.

The aggregation operation allows to reword the problem MOP as follows:

$$\text{(P')} \qquad \begin{cases} \min f_k(x) \\ g_j(x) \leq 0, j \in \overline{1:m} \\ f_i(x) \leq \varepsilon_i, i \in \overline{1:p}, i \neq k \\ x \in \mathbb{R}^n \end{cases},$$

where $\epsilon_i, i = \overline{1,p}, i \neq k$ are the parameters that must be determined before the looking for of the optimal solutions. Here, the objective function $f_k(x)$ is chosen randomly, but it is considered to be the prior objective function. For the following, let us set that $h_l = g_l, l = \overline{1,m},\ h_{m+i} = f_i - \epsilon_i, i = \overline{1,p}, i \neq k$.

The Lagrangian penalty function [8] is used at this step to transform the problem P' to an unconstrained problem. Thus, our initial problem becomes to minimize the following function:

$$\text{(Fp)} \qquad L(x,\epsilon,\eta) = f_k(x) + \eta \sum_{l=1}^{q} \Big(h_l(x) + |h_l(x)| \Big),$$

where $q = m+p-1$ and η is a large constant chosen such as $\eta \geq \dfrac{M - \max f_k(x)}{\sum_{l=1}^{m} h_l(x)}$.

MOGWAT algorithm is presented in Section 4.

3.2. **Convergence of MOGWAT.**

As the MOGWAT method presents a numerical algorithm, we have demonstrated its convergence in two steps: consistency results and stability results.

3.2.1. *Consistency.*

Definition 3.1. *The consistency of the numeric method is a property that ensures that the approached solutions converge to an exact solution for the initial problem when the discretization step is zero [14, 21].*

The following theorem is based on this definition.

Theorem 3.1. *Let us fix* $\epsilon \in \prod_{i=1}^{p} [\min f_i(x), \max f_i(x)]$. *MOGWAT method is consistency if and only if all optimal solution of problem (Fp) is a Pareto optimal solution of problem (MOP).*

Proof. Let x^* be an optimal solution of the problem (Fp) for a fixed point $\epsilon \in \prod_{i=1}^{p}[\min f_i(x), \max f_i(x)]$. Let us assume that x^* is not a Pareto optimal solution of problem (MOP). x^* being an optimal solution of the problem (Fp) means that for all other points $x \in \chi$, we have

$$L(x^*, \epsilon, \eta) \leq L(x, \epsilon, \eta). \tag{3.1}$$

That is equivalent to

$$\begin{aligned} &f_k(x^*) + \eta \sum_{j=1}^{m} \sum_{i=1, i\neq k}^{p} \Big(g_j(x^*) + f_i(x^*) - \varepsilon_i + |g_j(x^*) + f_i(x^*) - \varepsilon_i| \Big) \\ \leq\, &f_k(x) + \eta \sum_{j=1}^{m} \sum_{i=1, i\neq k}^{p} \Big(g_j(x) + f_i(x) - \varepsilon_i + |g_j(x) + f_i(x) - \varepsilon_i| \Big). \end{aligned}$$

Since x^* and x are admissible points, we have:

$$\begin{cases} g_j(x^*) \leq 0 \text{ and } g_j(x) \leq 0, \; \forall j = 1, \ldots, m \\ f_i(x^*) - \varepsilon_i \leq 0 \text{ and } f_i(x) - \varepsilon_i \leq 0, \; \forall i = 1, \ldots, p, \; i \neq k \end{cases}$$

By making the sum member per member of the two equations, we have: $g_j(x^*) + f_i(x^*) - \varepsilon_i \leq 0$ and $g_j(x) + f_i(x) - \varepsilon_i \leq 0$. That allows to obtain $g_j(x^*) + f_i(x^*) - \varepsilon_i + |g_j(x^*) + f_i(x^*) - \varepsilon_i| = 0$ and $g_j(x) + f_i(x) - \varepsilon_i + |g_j(x) + f_i(x) - \varepsilon_i| = 0$. That is also equivalent to

$$\begin{cases} \sum_{j=1}^{m} \sum_{i=1, i\neq k}^{p} \Big(g_j(x^*) + f_i(x^*) - \varepsilon_i + |g_j(x^*) + f_i(x^*) - \varepsilon_i| \Big) = 0 \\ \sum_{j=1}^{m} \sum_{i=1, i\neq k}^{p} \Big(g_j(x) + f_i(x) - \varepsilon_i + |g_j(x) + f_i(x) - \varepsilon_i| \Big) = 0 \end{cases}$$

Hence:

$$f_k(x^*) < f_k(x). \tag{3.2}$$

If x^* is not a Pareto optimal solution of problem (MOP) for a fixed ϵ then, there exists an optimal solution x such as: $f_i(x) \leq f_i(x^*), \; \forall i = \overline{1:p}, \; i \neq k$ and $\exists j \in \{1, 2, \ldots, p\}$ such as $f_j(x) < f_j(x^*)$. Two cases are possible:

1^{st} **case :** if $j = k$ then, $f_k(x) < f_k(x^*)$ and $f_i(x) \leq f_i(x^*) \leq \varepsilon_i, \forall i \neq k$, which contradicts the equation (3.2).

2^{nd} **case :** if $j \neq k$ then, $f_j(x) < f_j(x^*) \leq \varepsilon_j$ and $f_i(x) \leq f_i(x^*) \leq \varepsilon_i$, and $f_k(x) < f_k(x^*) \forall i \neq j, i \neq k$, which also contradicts the equation (3.2)

Therefore, x^* is a Pareto optimal solution of problem (MOP). □

3.2.2. *Stability.*

The stability of a numerical algorithm is a global property. It is a necessary quality in order to hope to obtain meaningful results. Stability is defined differently depending on the context. It refers to the spread of errors during the calculation of steps. It is the ability of the algorithm to avoid amplifying any deviations too much, to ensure the accuracy of the results obtained.

Iterative algorithms are stable if the perturbations of the numerical solutions do not increase with the number of iterations [21].

Definition 3.2. *[15, 19, 20] Let $\mathbb{E}$ be the mathematical expectation and $\mathbb{V}$ the variance of a random variable. Let us consider ALG, a numerical algorithm that has been defined by*

$$\text{(ALG)} \qquad \begin{cases} X(0) \textit{ given}, \\ X_i(t+1) = f(X_i(t)) \end{cases}.$$

Then

- ◇ *ALG is 1-order stable if for all agent of search i, $\lim\limits_{t->+\infty} \mathbb{E}\Big(X_i(t)\Big) < +\infty$.*
- ◇ *ALG is 2-order stable if for all agent of search i, $\lim\limits_{t->+\infty} \mathbb{V}\Big(X_i(t)\Big) = 0$.*

From the line 27 to 31 of the Algorithm **??**, we have formulated the numerical algorithm of MOGWAT as follows:

$$\begin{cases} X(0) \text{ given}, \\ X_i(t+1) = \sum\limits_{l=\alpha,\beta,\delta} \lambda_l g_{lj}(t) + \sum\limits_{l=\alpha,\beta,\delta} \lambda_l A_{lj} \big| C_{lj} g_{lj}(t) - x_{ij}(t) \big|, \ j = 1, \ldots, d; \ i = 1, \ldots, N. \end{cases}$$

Theorem 3.2. *MOGWAT Algorithm is 2-order stable.*

In the field of metaheuristic algorithm analysis, the stagnation hypothesis is a common one [19, 20]. In addition to this hypothesis of stagnation, let us consider the positions of the best research agents as constants.

Proof. Let us consider $X_i(t)$ positions of search agents as random variables. The position of $X_i(t+1)$, i^{th} agent of search at the iteration $t+1$ is in relation with $X_i(t)$ of iteration t^{th}, and the positions of G_l such as $l = \alpha, \beta, \delta$ three best research agents and also the random parameters A_{lj} and C_{lj}.

We have $A_{lj} \sim U\Big([-a(t), a(t)]\Big)$ and $C_{lj} \sim U\Big([0, 2]\Big)$. Then, the density function $f_{A_{lj}}$ of A_{lj} is given by $f_{A_{lj}}(x) = \dfrac{1}{2a(t)}$ and that of C_{lj} is given by $f_{C_{lj}}(x) = \dfrac{1}{2}$.

By using the transfer theorem, we have obtained:

$$\mathbb{E}(A_{lj}^r) = \int_{-a(t)}^{a(t)} \frac{x^r}{2a(t)} dx = \frac{(a(t))^{r+1} - (-a(t))^{r+1}}{2(r+1)a(t)} = \begin{cases} \dfrac{a(t)^r}{r+1}, r \text{ pair} \\ 0, \text{if else} \end{cases} .$$

With the same reasoning, we have gotten:

$$\mathbb{E}\Big(C_{lj}^r\Big) = \frac{2^r}{r+1}. \tag{3.3}$$

Elsewhere, we have:

$$\forall t,\ \mathbb{E}(x_{ij}(t)) = \mathbb{E}\Big(\sum_{l=\alpha,\beta,\delta} \lambda_l g_{lj}(t) + \sum_{l=\alpha,\beta,\delta} \lambda_l A_{lj} |C_{lj} g_{lj}(t) - x_{ij}(t)|\Big),$$

$j = 1, \ldots, d$; $i = 1, \ldots, N$.

As the mathematical expectation being linear and the variables A_{lj} and $|C_{lj}g_{lj} - x_{ij}|$ being independent, on have:

$$\begin{aligned} \mathbb{E}(x_{ij}(t)) &= \mathbb{E}\Big(\sum_{l=\alpha,\beta,\delta} \lambda_l g_{lj}(t)\Big) + \mathbb{E}\Big(\sum_{l=\alpha,\beta,\delta} \lambda_l A_{lj} |C_{lj} g_{lj}(t) - x_{ij}(t)|\Big) \\ &= \mathbb{E}\Big(\sum_{l=\alpha,\beta,\delta} \lambda_l g_{lj}(t)\Big) \text{ because } \mathbb{E}\Big(A_{lj}\Big) = 0 \\ &= \sum_{l=\alpha,\beta,\delta} \lambda_l G_l(t) < +\infty. \end{aligned}$$

On the one hand, the 2-order of moment can be calculated as follows:

$$\begin{aligned} \mathbb{V}\Big(x_{ij}(t+1)\Big) &= \mathbb{E}\Big(x_{ij}(t+1) - \mathbb{E}\Big(x_{ij}(t+1)\Big)\Big)^2 \\ &= \mathbb{E}\Big(x_{ij}(t+1) - \sum_{l=\alpha,\beta,\delta} \lambda_l G_l(t)\Big)^2 \end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}\Big(\sum_{l=\alpha,\beta,\delta} \lambda_l A_{lj} |C_{lj} g_{lj}(t) - x_{ij}(t)| \Big)^2 \\
&= \mathbb{E}\Big(\lambda_\alpha A_{\alpha j} |C_{\alpha j} g_{\alpha j}(t) - x_{ij}(t)| + \lambda_\beta A_{\beta j} |C_{\beta j} g_{\beta j}(t) - x_{ij}(t)| \\
&\quad + \lambda_\delta A_{\delta j} |C_{\delta j} g_{\delta j}(t) - x_{ij}(t)| \Big)^2 .
\end{aligned}
$$

By posing

$$
\begin{cases}
a = \lambda_\alpha A_{\alpha j} |C_{\alpha j} g_{\alpha j} - x_{ij}(t)| \\
b = \lambda_\beta A_{\beta j} |C_{\beta j} g_{\beta j} - x_{ij}(t)| \\
c = \lambda_\delta A_{\delta j} |C_{\delta j} g_{\delta j} - x_{ij}(t)|
\end{cases}
$$

Then, we have: $(a+b+c)^2 = a^2 + b^2 + c^2 + 2(ab + ac + bc)$.

As $\mathbb{E}\Big(a+b+c\Big)^2 = \mathbb{E}(a^2) + \mathbb{E}(b^2) + \mathbb{E}(c^2) + 2(\mathbb{E}(ab) + \mathbb{E}(ac) + \mathbb{E}(bc))$ by replacing each element with its expression and taking into account that $\mathbb{E}\Big(A_{lj}\Big) = 0, l = \alpha, \beta, \delta$, we have obtained:

$$
\begin{aligned}
&\mathbb{V}(x_{ij}(t+1)) \\
&= \mathbb{E}\Big((\lambda_\alpha A_{\alpha j} |C_{\alpha j} g_{\alpha j} - x_{ij}(t)|)^2 + (\lambda_\beta A_{\beta j} |C_{\beta j} g_{\beta j} - x_{ij}(t)|)^2 \\
&\quad + (\lambda_\delta A_{\delta j} |C_{\delta j} g_{\delta j} - x_{ij}(t)|)^2 \Big) \\
&= \mathbb{E}\Big((\lambda_\alpha A_{\alpha j}\Big)^2 \Big(C_{\alpha j} g_{\alpha j} - x_{ij}(t)\Big)^2 + \Big(\lambda_\beta A_{\beta j}\Big)^2 \Big(C_{\beta j} g_{\beta j} - x_{ij}(t)\Big)^2 + (\lambda_\delta A_{\delta j}\Big)^2 \\
&\quad \times \Big(C_{\delta j} g_{\delta j} - x_{ij}(t)\Big)^2 \Big) \\
&= \mathbb{E}\Big(\lambda_\alpha A_{\alpha j}\Big)^2 \mathbb{E}\Big(C_{\alpha j} g_{\alpha j} - x_{ij}(t)\Big)^2 + \mathbb{E}\Big(\lambda_\beta A_{\beta j}\Big)^2 \mathbb{E}\Big(C_{\beta j} g_{\beta j} - x_{ij}(t)\Big)^2 \\
&\quad + \mathbb{E}\Big(\lambda_\delta A_{\delta j}\Big)^2 \mathbb{E}\Big(C_{\delta j} g_{\delta j} - x_{ij}(t)\Big)^2 \\
&= \lambda_\alpha^2 \frac{a^2(t)}{3} \mathbb{E}\Big(C_{\alpha j} g_{\alpha j} - x_{ij}(t)\Big)^2 + \lambda_\beta^2 \frac{a^2(t)}{3} \mathbb{E}\Big(C_{\beta j} g_{\beta j} - x_{ij}(t)\Big)^2 \\
&\quad + \lambda_\delta^2 \frac{a^2(t)}{3} \mathbb{E}\Big(C_{\delta j} g_{\delta j} - x_{ij}(t)\Big)^2 \\
&= \lambda_\alpha^2 \frac{a^2(t)}{3} \mathbb{E}\Big(C_{\alpha j}^2 g_{\alpha j}^2 - 2x_{ij}(t) C_{\alpha j} g_{\alpha j} + x_{ij}^2(t)\Big) + \lambda_\beta^2 \frac{a^2(t)}{3} \mathbb{E}\Big(C_{\beta j}^2 g_{\beta j}^2 \\
&\quad - 2x_{ij}(t) C_{\beta j} g_{\beta j} + x_{ij}^2(t)\Big) + \lambda_\delta^2 \frac{a^2(t)}{3} \mathbb{E}\Big(C_{\delta j}^2 g_{\delta j}^2 - 2x_{ij}(t) C_{\delta j} g_{\delta j} + x_{ij}^2(t)\Big).
\end{aligned}
$$

Therefore, by using the expressions of $\mathbb{E}\Big(A^r_{lj}\Big)$, $\mathbb{E}\Big(C^r_{lj}\Big)$ and $\mathbb{E}(x_{ij}(t))$, we have obtained:

$$\begin{aligned}\mathbb{V}(x_{ij}(t+1)) = \frac{a^2(t)}{3}\Big(\frac{4}{3}\sum_{l=\alpha,\beta,\delta}\lambda_l^2 G_l^2 - 2\sum_{l=\alpha,\beta,\delta}\lambda_l^2 G_l(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l(t)) \\ + \sum_{l=\alpha,\beta,\delta}\lambda_l^2\mathbb{E}(x_{ij}^2(t))\Big).\end{aligned} \tag{3.4}$$

One the other hand, the 2-order of moment can be calculated as follows:

$$\begin{aligned}\mathbb{V}(x_{ij}(t)) &= \mathbb{E}\Big(x_{ij}(t) - \sum_{l=\alpha,\beta,\delta}\lambda_l G_l\Big)^2 \\ &= \mathbb{E}(x_{ij}^2(t)) - 2(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)\mathbb{E}(x_{ij}(t)) + (\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)^2 \\ &= \mathbb{E}(x_{ij}^2(t)) - 2(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l) + (\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)^2 \\ &= \mathbb{E}(x_{ij}^2(t)) - (\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)^2.\end{aligned}$$

From the last equation, we have $\mathbb{E}(x_{ij}^2(t)) = \mathbb{V}(x_{ij}(t)) + (\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)^2$. By using this in 3.4, we have:

$$\begin{aligned}\mathbb{V}(x_{ij}(t+1)) = \frac{a^2(t)}{3}\Big(\sum_{l=\alpha,\beta,\delta}\lambda_l^2\mathbb{V}(x_{ij}(t)) + \sum_{l=\alpha,\beta,\delta}\lambda_l^2(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)^2 + \frac{4}{3}\sum_{l=\alpha,\beta,\delta}\lambda_l^2 G_l^2 \\ - 2(\sum_{l=\alpha,\beta,\delta}\lambda_l^2 G_l))(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l))\Big).\end{aligned}$$

By setting

$$\begin{cases}\mathbb{V}_{t+1} &= \mathbb{V}\Big(x_{ij}(t+1)\Big) \\ \mathbb{V}_t &= \mathbb{V}\Big(x_{ij}(t)\Big) \\ b_t &= \dfrac{a^2(t)}{3} \\ \gamma &= \sum_{l=\alpha,\beta,\delta}\lambda_l^2 \\ p_0 &= \sum_{l=\alpha,\beta,\delta}\lambda_l^2\Big(\sum_{l=\alpha,\beta,\delta}\lambda_l^2(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)^2 + \dfrac{4}{3}\sum_{l=\alpha,\beta,\delta}\lambda_l^2 G_l^2 \\ & \quad -2(\sum_{l=\alpha,\beta,\delta}\lambda_l^2 G_l)(\sum_{l=\alpha,\beta,\delta}\lambda_l G_l)\Big)\end{cases}$$

we have $p_0 \geq 0$. That allows to reword equation 3.4 as follows:

$$\mathbb{V}_{t+1} = b_t\Big(\gamma \mathbb{V}_t + p_0\Big). \tag{3.5}$$

Equation 3.5 can be considered as a first order dynamic system and can be solving as follows:

$$\begin{aligned}
\mathbb{V}\Big(x_{ij}(t)\Big) &= \frac{1}{\gamma}\Big(\mathbb{V}(x_{ij}(1))\prod_{\tau=1}^{t-1} b_\tau + p_0 b_{t-1}\Big(1+\sum_{\tau=1}^{t-2}\prod_{\zeta=\tau}^{t-2} b_\tau\Big)\Big),\ t>1\\
&= \frac{1}{\gamma}\Big(\mathbb{V}(x_{ij}(1))\prod_{t=1}^{T-1} b_t + p_0 b_{T-1}\Big(1+\sum_{t=1}^{T-2}\prod_{\zeta=t}^{T-2} b_\tau\Big)\Big)\\
&< \frac{1}{\gamma}\Big(\mathbb{V}(x_{ij}(1))b_{T-1} + p_0 b_{T-1}\Big(1+\sum_{t=1}^{T-2} 1\Big)\Big)\\
&< \frac{4}{3\gamma T^2}\Big(\mathbb{V}(x_{ij}(1)) + p_0(T-1)\Big).
\end{aligned}$$

As $\mathbb{V}\Big(x_{ij}(1)\Big) < +\infty$, then

$$\lim_{T->+\infty} \mathbb{V}(x_{ij}(T)) \leq \lim_{T->+\infty} \frac{4}{3\gamma T^2}\Big(\mathbb{V}(x_{ij}(1)) + p_0(T-1)\Big) = 0$$

and $\lim_{t->+\infty} \mathbb{V}\Big(x_{ij}(t)\Big) = 0$. □

3.3. **Numerical results.**

To evaluate the numerical performance of our metaheuristic method a comparative has been done with two other metaheuristic methods taking into the literature [12, 13]. This comparison has been made on the convergence and distribution of the obtained Pareto optimal solutions by method. Therefore, in this section, we have, at first, presented the performance parameters. Then, we have given the using test problems in the table 1. After that, we have provided two tables for giving the performance parameters, and finally, we do some comments.

3.3.1. *Performance parameters.*

Distance générationnelle γ: For a numerical method, it is important to find solutions that are closest to the true Pareto front. The parameter γ values allows us to assess this property. It evaluates the errors caused by taking the obtained solutions as the analytic solutions. In other words, the level of convergence of the method is measured by γ. More it is next to zero, the method is good [16]. The following formula is used to calculate this parameter.

$$\gamma = \frac{1}{n}\left(\sum_{i=1}^{n} d_i^p\right)^{\frac{1}{p}}. \tag{3.6}$$

In this paper, we have fixed $p = 2$. Notice that d_i is the distance between the i^{th} solutions of the obtained solutions and the analytic solutions. n is the number of obtained solutions.

Spread Δ : For a numerical method, it is important to find solutions that cover the Pareto front uniformly. The parameter Spread Δ allows evaluating of this property. It provides a measure of the distribution of the obtained solutions. When its values are close to zero, the distribution of the method is good [17]. This parameter is computed by the following formula:

$$\Delta = \frac{\sum_{m=1}^{M} d_m^e + \sum_{i=1}^{|Q|} |d_i - \overline{d}|}{\sum_{m=1}^{M} d_m^e + |Q|\overline{d}}$$

with d_i the Euclidean distances between two related solutions with the average value $\overline{d}$; And d_m^e is the distance between extreme solutions.

3.3.2. *Test problems.*

Six test problems have been chosen that are well-known and used in the literature [2, 4, 8]. These problems have already been dealt with by the methods MSSA [13] and the MOGOA [12].

Table 1: Test problems

$\begin{cases} \textbf{FON} \\ \min f_1(x) = 1 - \exp(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}})^2\right) \\ \min f_2(x) = 1 - \exp(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}})^2\right) \\ 4 \leq x_i \leq 4, n = 10 \end{cases}$	$\begin{cases} \textbf{ZDT1} \\ \min f_1(x) = x_1 \\ \min f_2(x) = g(x) \times \left(1 - \sqrt{\frac{f_1(x)}{g(x)}}\right) \\ g(x) = 1 + \frac{9}{n-1} \times \sum_{i=2}^{n} x_i \\ 0 \leq x_i \leq 1, i = \overline{1:30} \end{cases}$

ZDT1-linear $\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x) \times \left(1 - \dfrac{f_1(x)}{g(x)}\right) \\ g(x) = 1 + \dfrac{9}{n-1} \times \sum_{i=2}^{n} x_i \\ 0 \leq x_i \leq 1, i = \overline{1:30} \end{cases}$	**ZDT2** $\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x) \times \left(1 - \left(\dfrac{f_1(x)}{g(x)}\right)^2\right) \\ g(x) = 1 + \dfrac{9}{n-1} \times \sum_{i=2}^{n} x_i \\ 0 \leq x_i \leq 1, i = \overline{1:30} \end{cases}$
ZDT3 $\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x) \times h(x) \\ g(x) = 1 + \dfrac{9}{n-1} \times \sum_{i=2}^{n} x_i \\ h(x) = 1 - \sqrt{\dfrac{f_1(x)}{g(x)}} - \dfrac{f_1(x)}{g(x)} \sin\left(10\pi f_1(x)\right) \\ 0 \leq x_i \leq 1, i = \overline{1:30} \end{cases}$	**ZDT4** $\begin{cases} \min f_1(x) = x_1 \\ \min f_2(x) = g(x) \times \left(1 - \sqrt{\dfrac{f_1(x)}{g(x)}}\right) \\ g(x) = 1 + 10(n-1) \\ \quad + \sum_{i=2}^{n} (x_i^2 - 10\cos(4\pi x_i)) \\ 0 \leq x_1 \leq 1 - 5 \leq x_i \leq 5, i = \overline{2:10} \end{cases}$

The following figures show, for each test problem, a graphic representation of the obtained solutions by the MOGWAT method and the analytical Pareto front.

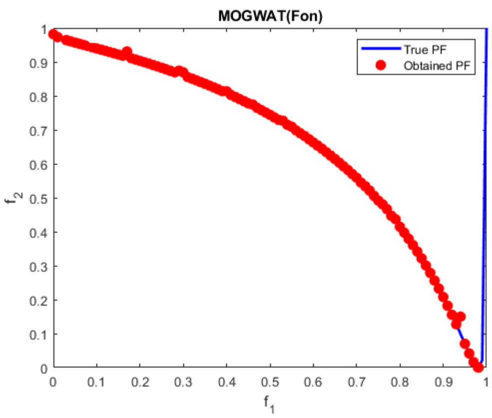

FIGURE 3. Optimal front obtained by MOGWAT on FON problem

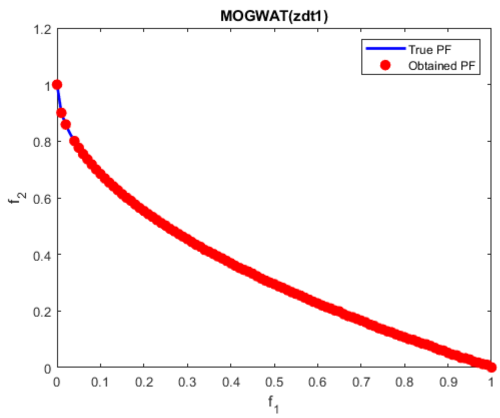

FIGURE 4. Optimal front obtained by MOGWAT on ZDT1 problem

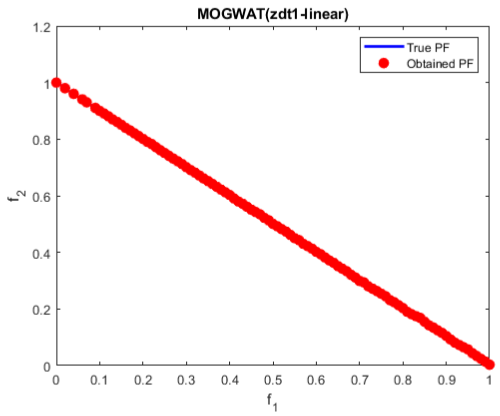

FIGURE 5. Optimal front obtained by MOGWAT on ZDT1-linear

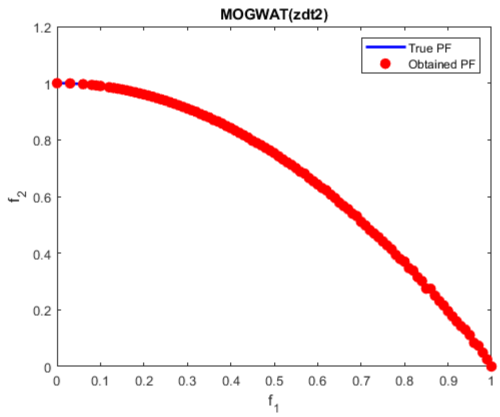

FIGURE 6. Optimal front obtained by MOGWAT on ZDT2

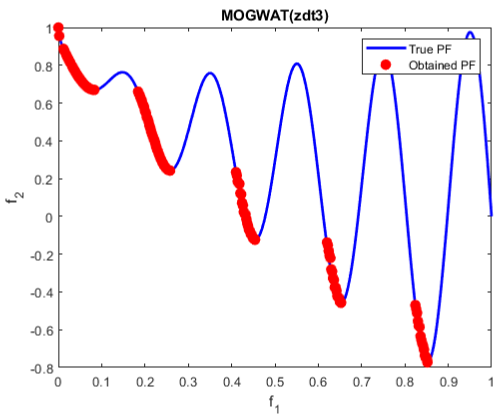

FIGURE 7. Optimal front obtained by MOGWAT on ZDT3

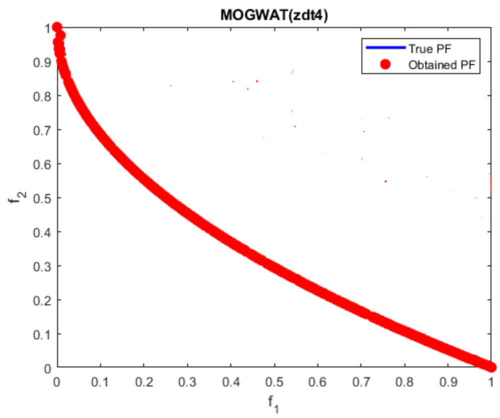

FIGURE 8. Optimal front obtained by MOGWAT on ZDT4

3.3.3. *Table of comparison.*

Since the three methods are stochastic, in order to obtain the solution to each problem, we must run the algorithm thirty times and then calculate the mean and the standard deviation. The following tables have been created using this process.

TABLE 2. Values of parameter γ for the three methods

γ	MOGWAT		MSSA		MOGOA	
	mean	std	mean	std	mean	std
FON	0,000420	8,8077e-5	0,00046	2,7308e-5	0,00016	0,00040
ZDT1	0,000130	1,3744e-5	0,01428	0,00221	0,00132	0,00711
ZDT1-linear	0,000270	1,6664e-5	0,01842	0,00164	0,00109	0,00592
ZDT2	0,000370	2,2829e-5	0,01748	0,00405	0,00145	4,5983e-5
ZDT3	0,000120	4,9890e-6	0,00472	0,00079	0,00014	0,00079
ZDT4	0,003915	0,000490	0,03411	0,00439	0,00739	0,01017

TABLE 3. Values of parameter Δ for the three methods

Δ	MOGWAT		MSSA		MOGOA	
	mean	std	mean	std	mean	std
FON	0,59468	0,00914	1,07940	0,03238	0,11442	0,34326
ZDT1	0,25853	0,05011	1,22520	0,04682	0,03678	0,19812
ZDT1-linear	0,10850	0,02108	1,14454	0,04906	0,04043	0,21777
ZDT2	0,24873	0,01179	1,12675	0,05610	0,03448	0,18572
ZDT3	0,36174	0,00559	1,20691	0,08501	0,03469	0,18684
ZDT4	0,27140	0,02020	1,05515	0,05817	0,12229	0,36666

3.3.4. *Comments.*

- ◇ From the table reftab1, it can be observed that the solutions provided by the MOGWAT method are all close to zero. It indicates that the method has good convergence. Furthermore, it has the bested value compared to two other methods. Therefore, we can conclude that MOGWAT is better than MSSA and MOGOA in terms of convergence.
- ◇ An analysis of the results presented in table 3.3.3 proves that MOGAT has a good distribution of obtained solutions because the values of parameters γ are all next to zero. MOGWAT is not the best method for all test problems, but it is not dominated by the two other methods in terms of distribution.

On the whole, MOGWAT is better than MSSA and MOGOA for these six test problems that have been dealt with in the paper.

4. The algorithm

Algorithm 1: Multiobjective Optimizer based on the Grey Wolf Attack technics

Data: $N,\ d,\ lb,\ ub,\ T,\ pm$

for $i = 1, 2, 2, \cdots, p$ **do**
  compute $f_i^l = \min f_i(x)$
  compute $f_i^u = \max f_i(x)$
for $i = 1, 2, 3, \cdots$ **do**
  choose $\epsilon_i \in [f_i^l, f_i^u]$
  intialize $P_1 = \{X_i(1), i = 1, \ldots, N\}\} = \{X_1(1), X_2(1), \ldots, X_N(1)\}$
  t=1
  while $t<T$ **do**
    for $i = 1, 2, \cdots, N$ **do**
      compute $\gamma = rand(1)$
      select $Y_i(t)$
      compute $p_i(t) = \dfrac{L(X_i(t), \epsilon, \eta)}{\sum_{i=1}^{N} L(X_i(t), \epsilon, \eta)} < \gamma$
    for $i = 1, 3, 5, \cdots, N$ **do**
      compute $a = rand(1)$
      $Q_t = \{a \times Y_i(t) + (1-a) \times Y_{i+1}(t), (1-a) \times Y_i(t) + a \times Y_{i+1}(t)\}$
    $R_t = P_t \cup Q_t$
    for $i = 1, 2, \cdots, 2N$ **do**
      each $Z_i(t) \in R_t$
      Generate $\zeta = rand(1),\ \omega = randn(1)$
      if $\zeta < pm$ **then**
        $Z_i(t) = Z_i(t) + \omega$
    for $i = 1, 2, \cdots, 2N$ **do**
      compute $L(Z_i(t), \epsilon_i, \eta)$
    Reordering such as
    $L(Z_{\sigma(1)}(t), \epsilon, \eta) < L(Z_{\sigma(2)}(t), \epsilon, \eta) < \cdots < L(Z_{\sigma(2N)}(t), \epsilon, \eta)$
    $R_t^{new} = \{Z_{\sigma(1)}(t), Z_{\sigma(2)}(t), \ldots, Z_{\sigma(N)}(t)\}$
    Set $G_\alpha(t) = Z_{\sigma(1)}(t),\ G_\beta(t) = Z_{\sigma(2)}(t),\ G_\delta(t) = Z_{\sigma(3)}(t)$
    for $i = 1, 2, \cdots, 2N$ **do**
      compute $D_\alpha^i(t) = |C_\alpha G_\alpha(t) - Z_i(t)|, V_\alpha^i(t) = G_\alpha(t) - A_\alpha D_\alpha^i(t)$
      compute $D_\beta^i(t) = |C_\beta G_\beta(t) - Z_i(t)|, V_\beta^i(t) = G_\beta(t) - A_\beta D_\beta^i(t)$
      compute $D_\delta^i(t) = |C_\delta G_\delta(t) - Z_i(t)|, V_\delta^i(t) = G_\delta(t) - A_\delta D_\delta^i(t)$
    $P_{t+1} = \{\lambda_\alpha V_\alpha^i(t) + \lambda_\beta V_\beta^i(t) + \lambda_\delta V_\delta^i(t), i = 1, \ldots, N, \sum_{l=\alpha,\beta,\delta} \lambda_l = 1\}$
    $t \leftarrow t + 1$
  Return $G_\alpha(T)$ as the optimal solution

5. Conclusion

This paper focused on the performance study of the MOGWAT method, which has been proposed in previous works. First, we proposed the description of the steps of the method. Then, we have presented the algorithm for the methods. Finally, we proposed the theoretical and numerical performance of the method. Concerning the theoretical performance, we have demonstrated the convergence of the method through three theorems. Regarding the numerical performance, six test problems have been addressed, and the outcomes have been compared to those of MSSA and MOGOA method. According to the obtained results, MOGWAT is the best choice for the resolution of multiobjective optimization problems.

For our next work, we will investigate the complexity of the method in order to improve the distribution of solutions and then its applications for the resolution of real-world problems.

6. Conflict of Interests

The authors declared that there is no conflict of interests in their submitted paper.

References

[1] M. Parvizi, E. Shadkam, N. jahani: *A hybrid COA/ε-constraint method for solving multiobjective problems*, International Journal in Foundations of Computer Science and Technology (IJFCST). **5**(5) (2O14), 27–40.

[2] M. Premkumar, P. Jangir, R. Sowmya, H.H. Alhelou, A.A. Heidari, H. Chen: *MOSMA: Multi-Objective Slime Mould Algorithm Based on Elitist Non-Dominated Sorting*, IEEE Access, **9** (2021), 3229–3248.

[3] S. Alexandre, S. Kounhinir, C. Abdoulaye, S. Blaise: *Effect of Aggregation Function in MOMA-Plus Method For Obtaining Pareto Optimal Solutions*, Applications and Applied Mathematics: An International Journal (AAM). **17**(2), (2022), 536-557.

[4] A. Sadollaha, H. Eskandar, J.H. Kima: *Water cycle algorithm for solving constrained multi-objective optimization problems*, Applied Soft Computing, **27** (2015), 279–298.

[5] D. Wolpert, W. Macready: *No free lunch theorems for optimization*, IEEE Trans Evol Comput, **1** (1997), 67–82.

[6] S. MIRJALILI, S. SAREMI, S.M. MIRJALILI, LEANDRO DOS S. COELHO: *Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization*, Elsevier, Expert Systems With Applications, **47** (2016), 106–119.

[7] S. MIRJALILI, P. JANGIR, S. SAREMI: *Multi-objective ant lion optimizer: a multi-objective optimization algorithm for solving engineering problems*, The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies, **46** (2017), 79–95.

[8] W. BAMOGO, K. SOME, J. PODA: *Grey wolves attack process for the Pareto optimal front construction in the multiobjective optimization*, Europan journal of pure and applied mathematics, **16**(1) (2023), 595–608.

[9] W.O. SAWADOGO, P.O.F. OUÉDRAOGO, K. SOMÉ, N. ALAA, B. SOMÉ: *Modified hybrid grey wolf optimizer and genetic algorithm (HmGWOGA) for global optimization of positive functions*, Advances in Differential Equations and Control Processes, **20**(2) (2019), 187–206.

[10] S. MIRJALILI, S.M. MIRJALILI, A. LEWIS: *Grey wolf optimizer*, Advances in Engineering Software, **69** (2014), 46–61.

[11] K. DEB, S. AGRAWAL, A. PRATAP, T. MEYARIVAN: *A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization: NSGA-II*, In International Conference on Parallel Problem Solving from Nature, **6** (2002), 182–197.

[12] S.Z. MIRJALILI, S. MIRJALILI, S. SAREMI, H. FARIS, I. ALJARAH : *Grasshopper optimization algorithm for multi-objective optimization problems*, Springer, Applied Intelligence, **48** (2018), 805–820.

[13] S. MIRJALILI, A.H. GANDOMI, S.Z. MIRJALILI, S. SAREMI, H. FARIS, S.M. MIRJALILI: *Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems*, Elsevier, Advances in Engineering Software, **00** (2017), 1–29.

[14] E.G. DA SILVA: *Méthodes et analyse numériques*, Institut polytechnique de Grenoble, Engineering school, 2005.

[15] M. JIANG, Y.P. LUO, S.Y. YANG: *Stochastic convergence analysis and parameter selection of the standard particule swarm optimization algorithm*, Elsevier, Information processing letters **102** (2007), 8–16.

[16] O. SCHUTZE, X. ESQUIVEL, A. LARA, C.A. COELLO: *Using the Averaged Hausdorff Distance as a Performance Measure in Evolutionary Multiobjective Optimization*, IEEE Transactions on evolutionary computation, **16**(2012), 504–522.

[17] Y.N. WANG, L.H. WU, X.F. YUAN: *Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure*, Soft Comput, **14**(2010), 193–209.

[18] C. MURO, R. ESCOBEDO, L. SPECTOR, R. COPPINGER: *Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations*, Behavioural processes, **88**(2011), 192-197.

[19] R. POLI: *Mean and Variance of the Sampling Distribution of Particle Swarm Optimizers During Stagnation*, IEEE Transactions on Evolutionary Computation, **13**(2009), 712-721.

[20] V. KADIRKAMANATHAN, K. SELVARAJAH, P.J. FLEMING: *Stability analysis of the particle dynamics in particle swarm optimizer*, IEEE Transactions on Evolutionary Computation, **10**(2006), 245-255.

[21] S.A. BROMA: *Implementation de l'equation de Black-Scholes en Python par la methode de difference finie* , Université Panthéon Sorbonne, Paris, 2019.

[22] G. COCCHI, M. LAPUCCI: *An augmented Lagrangian algorithm for multiobjective optimization,* Computational Optimization and Applications, **77**(2020), 29-56.

[23] E.G. BIRGIN, J.M. MARTINEZ : *Practical Augmented Lagrangian Methods for Constrained Optimization,* Society for industrial and Applied Mathematics, SIAM, (2014), .

[24] Z. AMIDOU, S. KOUNHINIR, J. JOSEPH: *Obtaining Optimal Pareto Solutions using a Hybrid Approach combining ϵ-constraint and MOMA-Plus Method*, International Journal of Numerical Methods and Applications. **23**(1) (2023), 107-129.

[25] E.G. BIRGIN, J.M. MARTINEZ : *On the application of an augmented Lagrangian algorithm to some portfollio problems,* EURO Journal on Computational Optimization, **4** (2016), 79-92.

DÉPARTEMENT DE MATHÉMATIQUES ET INFORMATIQUE
UNIVERSITÉ NORBERT ZONGO
RN14, KOUDOUGOU,
BUKRINA FASO.
Email address: menabamogo@gmail.com

DÉPARTEMENT DE MATHÉMATIQUES ET INFORMATIQUE
UNIVERSITÉ NORBERT ZONGO
RN14, KOUDOUGOU,
BUKRINA FASO.
Email address: sokous11@gmail.com

DÉPARTEMENT D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE
UNIVERSITÉ DE CALAVI
ROUTE DE DANGBO, PORTO-NOVO,
BENIN.
Email address: gdegla@gmail.com