Periodicals of Engineering and Natural Sciences ISSN 2303-4521
Vol. 9, No. 2, March 2021, pp.400-409

A hybrid Grey Wolf optimizer with multi-population differential evolution for global optimization problems

Nuha Sami Mohsin [1], Buthainah F. Abd[2], Rafah Shihab Alhamadani[3]
[1]Faculty of Education Ibn Rushd for Human Sciences, Baghdad University, Baghdad, Iraq
[2]Smart Cities Dept, University of Information Technology and Communications, Baghdad, Iraq
[3]Iraqi Commission for Computers & Informatics, Informatics Institute for Postgraduate Studies, Baghdad, Iraq

ABSTRACT

The optimization field is the process of solving an optimization problem using an optimization algorithm. Therefore, studying this research field requires to study both of optimization problems and algorithms. In this paper, a hybrid optimization algorithm based on differential evolution (DE) and grey wolf optimizer (GWO) is proposed. The proposed algorithm which is called "MDE-GWONM" is better than the original versions in terms of the balancing between exploration and exploitation. The results of implementing MDE-GWONM over nine benchmark test functions showed the performance is superior as compared to other stat of arts optimization algorithms

Keywords: Optimization, Metahuristics , Grey Wolf Optimizater, Differential Evolution, Multi-Population

Corresponding Author:

Nuha Sami Mohsin,
Faculty of Education Ibn Rushd for Human Sciences,
Baghdad University,
Baghdad, Iraq,
Email: nuha.sami@ircoedu.uobaghdad.edu.iq

1. Introduction

Optimization problems are problems that form a unique class of problems while trying to either minimize or maximize the mathematical function of several variables with respect to certain constraints. This general framework can be used to model several problems, both theoretical and real-world. The structure of mathematical models (or mathematical programming model) can generally be represented as follows [1]–[3]:

$$\begin{aligned} &Max\ or\ Min\ f(x_i), && (i = 1, 2, 3, \dots, D) \\ &subject\ to\ \ h_j(x_i) = 0, && (j = 1, 2, 3, \dots, J), \\ &\qquad g_k(x_i) \leq 0, && (k = 1, 2, 3, \dots, K) \end{aligned} \tag{1}$$

where x is the decision variables, and $f_i(x), h_j(x),$ and $g_k(x)$ are functions of the design vector.

$$\boldsymbol{x} = \ (x_1, x_2, x_3, \dots, x_D)^T \tag{2}$$

Metaheuristics have been commonly used in the field of optimization compared to other methods due to the simplicity and robustness of their outcome when used in several fields. Several studies have been conducted in the area of metaheuristics, including the introduction of novel methods, performance evaluations and applications [4]–[6]. Meanwhile, it is still believed that the field of metaheuristics is yet to mature compared to mathematics, physics, or chemistry [7]. With time, several studies are anticipated on the issues facing metaheuristic computing.

Metaheuristics provide solutions to optimization problems by searching for the best approach to a given problem [8]. This solution search can be executed using several agents which actually form a pool of emerging solutions during multiple iterations based on a set of mathematical principles or equations. The iterations are constantly executed until a solution that meets most of the set criteria is found. This optimum solution is considered as the best achieved solution, and such a system is said to have converged. On the contrary, heuristic methods attain near-optimal solutions with less computation time, but they mostly depend on the problem[9].

In general, metaheuristics could be classified into two main classes. First, evolutionary algorithms where these algorithms depend on different evolutionary operators such as crossover, and mutation. Whil the second class is swarm intelligence algorithms, where several search agents try to find better solutions for a given optimization problem, both of these types belong to a general class called "Nature-Inspired Algorithm". There are tens of nature-inspired algorithm proposed during the last few decades, such as Particle Swarm Optimization (PSO), Aritificial Bee Coloney (ABC), Ant Colony Optimizer (ACO), Firefly Algorithm (FA), Cuckoo Search Algorithm (CSA), Ant Linor Optimizer (ALO), Bat Algorithm (BA), Nomadic Peaple Optimizer (NPO), Socio Evolution & Learning Optimization Algorithm (SELO), Teaching–Learning-based Optimization (TLBO), and Whale Optimization Algorithm (WOA)[10]–[19]. Differential Evolution (DE) is a common evolutionary algorithm, while grey wolf optimizer (GWO) is a common recent swarm intelligence algorithm[20], [21]. Nature inspired algorithms have been used for solving global optimization problems [22]–[26], data clustering [27], [28], machine learning [29]–[31], data security [32], [33] , and engineering problems[34]–[38] .

In this paper, DE is enhanced using a new multi-population arhcitucture, and hybridized with GWO for enhancing the balancing between the global and local search capabilities.

The rest of the paper is structured as follows: Section 2 presents an overview on DE, while Section 3 presents the proposed algorithm. Section 4 presents the results and discussion. Finally, Section 5 concludes the research.

2. An overiew on differential evolution (DE)

The DE algorithm for unconstrained global optimization is presented in this section but before proceeding with the detailed description of this algorithm, it is necessary to consider some basic things required of a global optimization solver and how such requirements will be met by the DE algorithm. Before deciding the optimizer for any application, it is important to consider some basic attributes of the optimizer[39]. Among these attributes of a solver that requires consideration are[40]:

- Generality: The solver must exhibit some levels of insensitivity to the structure of the underlying problem so that it can be generalizable to a larger problem domain.
- Reliability: The optimizer should be capable of finding the global optimum with an acceptable level of accuracy.
- Efficiency: The solvers computational complexity should ensure that the algorithm is suitable for small to moderate problems, such as problems with up to 100 dimensions.
- Ease of use: The algorithm should be easy to implement and understand. The number of parameters should also be limited so that too much fine tuning is not required for the algorithm to perform well.

Having considered these requirements, it seems the DE algorithm is among the appealing optimizers that can be considered global optimizers. DE was first introduced by Storn & Price [20] but has been subjected to several evaluations and modification aimed at improving its applicability and performance. The DE algorithm has demonstrated its efficiency and robustness as an evolutionary algorithm.

Descriptively, the DE algorithm is an evolution-based stochastic optimizer that executes functions by exploring the solution space using a set or population $S = \{x_1, x_2, \ldots, x_N\}$ of possible points/solutions just like the other evolutionary algorithms. The population size (denoted by the value N) is fixed throughout. The algorithms aim at the creation of a new population at each iteration by replacing some of the points in the existing population S with better points. As such, the population simply becomes a set of points $x_{i,g}$, in which i is the index of the population members while g is the iteration that the population belongs. Each $x_{i,g}$ is comprised of n components,

where n represents the problems' dimension. The population S is guided toward the global minimum via repeated reproduction (mutation & crossover) and selection processes.

The DE algorithm is similar to any other evolutionary algorithm, contains evolutionary operators (i.e., crossover, and mutation). The main steps of DE are given in the following pseducode [40]–[42].

Algorithm 1 DE Algorithm

Inputs Parameters:

$Populations\ Size(P)$,$Crossover(C_r)$, $Mutation\ Factors(F)$, $Stop\ Condition$

Procedure:

1. Define the objective function $f(X)$
2. Generate the initial population randomly
3. Initialize the population of the algorithm randomly
4. Evaluate each solution in the population using $f(X)$
4. *WHILE Stop Condition is Not Met*
5. FOR each solution X in the population
6. Apply the **Mutation** operator
7. Apply the **Crossover** operator
8. END FOR
9. Update the population using the **Selection** operator
10. *LOOP*
10. Determine and Return Best solution

3. The proposed algorithm

In this section, the main proposed algorithm is explained in details. It is called "Multi-Population Differential Evolution Grey Wolf Optimizer with Nelder Mead (MDE-GWONM)", which consists of three main stages, and each stage consists of several main steps. These stages and steps are explained as follows:

Stage 1: Initialization

This is the first stage, where the main information required for starting and executing the proposed algorithm are read, and the populations are initialized. This stage consists of the following steps:

Step 1 – Inputs: In this step, all the inputs for the proposed algorithm are read, such as the number of populations (for MP-DE), total number of iterations ($\#MaxItr$), number of search agents (for GWO), and the information of the optimization problem.

Step 2 – Initialize the Populations: In this work, DE algorithm is enhanced by using several populations instead of one in order enhance the global search ability of the algorithm. Therefore, in this step, all of the populations are initialized randomly. This step is explained in more details in Subsection 3.6.

Step 3 – Evaluate all solution using the optimization problem: in this step, the initialized candidate solutions in each population via the previous step are evaluated using the objective function, in order to identify the fitness value for each solution. It is important to mention that in this work, there are three different types of optimization problems, single-objective optimization problem, multi-objective optimization problem, and the workflow scheduling optimization problem. Based on the type which should be determined in the first step, the optimization problem should be used for the evaluation purpose.

Step 4 – Starting the main loop: In this step, the iterations are started for executing the searching process, until the stop condition is satisfied. Both modified algorithms, MP-DE and EGWO are executed inside the main loops. However, MP-DE algorithm is executed for the first 30% iterations only based on the following:

$$Algorithm = \begin{cases} MP-DE & if\ T \leq 0.3 * MaxItr \\ EGWO & Otherwise \end{cases} \quad (3.14)$$

Where T represents the current iteration, while $MaxItr$ represents the total number of iterations. It can be seen that MP-DE is executed when the current iteration is still less than 30% of the total number of iterations, meaning that, the proposed algorithm explores the search space using MP-DE for 30% of the whole algorithm, while EGWO is executed and exploited the search space for the rest 70% based on the generated solutions using MP-DE.

Stage 2: MP-DE

In this stage, the proposed modified DE algorithm, which is called "Multi-Population Differential Evolution (MP-DE)" is executed. MP-DE algorithm is responsible of exploring the search space, and used initialization step for EGWO algorithm. Meaning that, the resulted solutions of MP-DE are used as the first solutions for EGOW, then EGWO is executed for exploiting the search space and enhancing the results. MP-DE algorithm is explained in details in the next section.

Stage 3: EGWO

In this stage, the modified version of GWO which is called "Enhanced GWO (EGWO)" is executed for exploiting the search space and enhancing the positions for reaching better near optimal solutions. EGWO is a enhanced version of GWO by updating the parameter a based on four different values: start value, end value, total number of iterations, and the current iteration. The value of a is decreased linearly based on these four values, which means it starts large, then it decreases until it becomes at the end value. In other words, the value a effects on the searching performance, when a is large, meaning that the positions of the search agents are far from the current alpha, while when a is small, the positions of the search agent get closer to the current alpha. In addition to the previous modification, GWO algorithm is hybridized with NM algorithm, the local search ability of the algorithm should be enhanced when the search agents move towards the best solutions (W_{alpha}, W_{beta}, and W_delta).

Although EGWO is enhanced, however, EGWO still suffers from the initialization stage, when each search agent is initialized randomly via uniform distribution method in the search space. The drawback is that the random positions generated in this step may cover a wrong area in the search space, which causes trapping in the local minima in some cases. On the other side, DE algorithm in general as any other evolutionary algorithm performs well on the exploration side than the exploitation side, which may leads to slowdown the search process when the solutions trying to find better positions near to the optimal solutions.

In order to overcome the above-mentioned issues, a high-level hybrid algorithm between both of the algorithms is proposed. In other words, the proposed MP-DE algorithm (Stage 2) is used at the beginning of the algorithm for generating better solutions with good positions in the search space. Then, EGWO (Stage 3) is applied on the final generated solutions which performs the local search. Figure 1 and Figure 2 illustrate the block diagram and the flowchart of the proposed algorithm respectively.

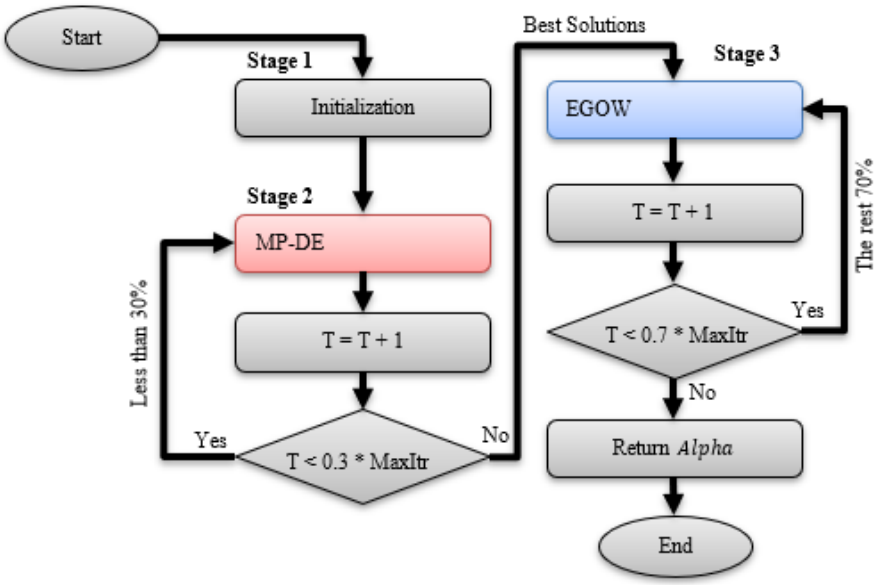


Figure 1 Block Diagram of the proposed algorithm

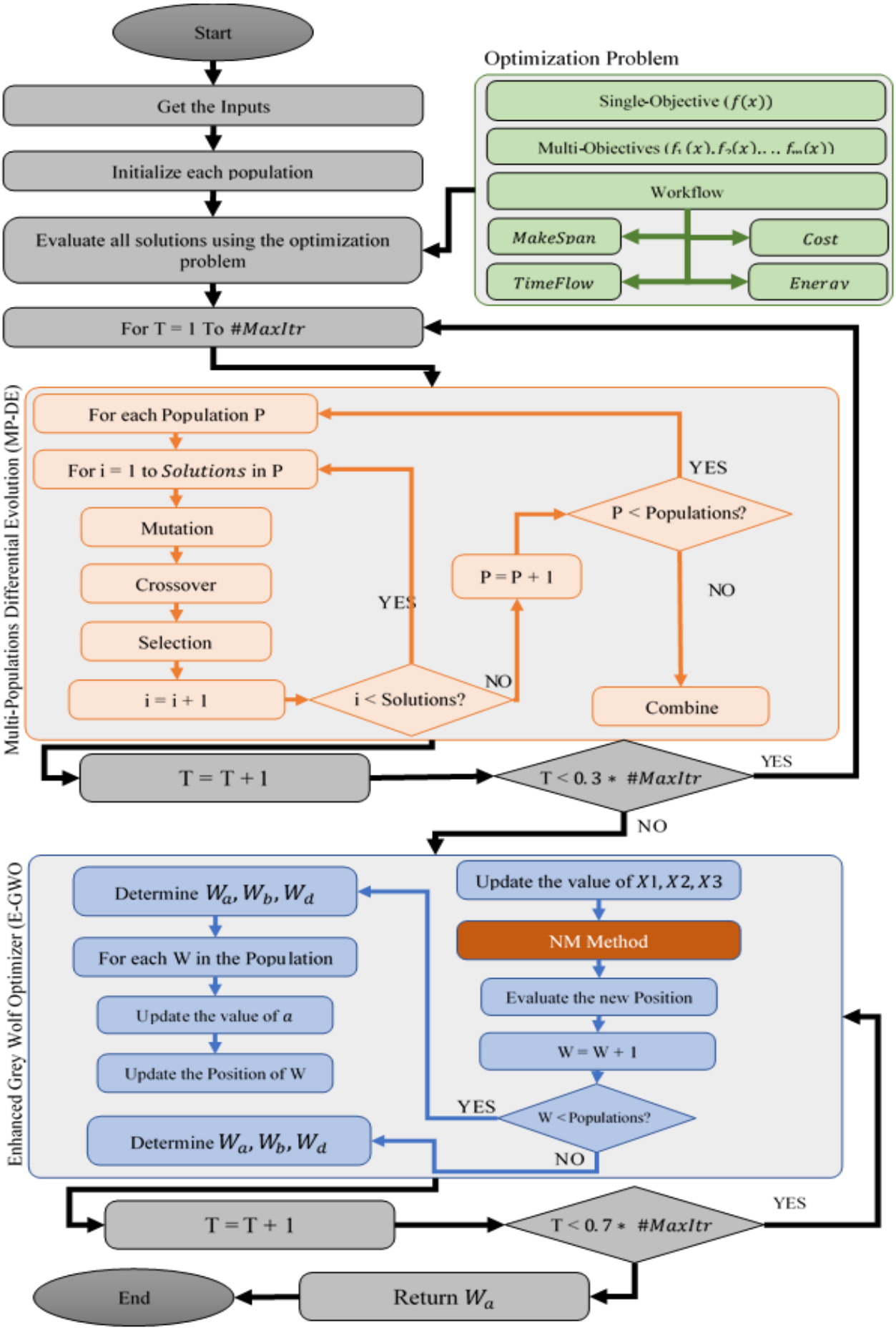


Figure 2 Flowchart of the Proposed Approach

4. Results and Discussion

This section presents the results of the proposed algorithms, after executing and recording all the experiments over the 9 benchmark test functions, which are presented in Table 1 below. The outcomes showed that the proposed approaches exerted superior performance and could reach the optimal solution for most test functions. The benchmark results and comparison are presented in Table 2, where it presents the comparison between the proposed algorithm and the original versions of GWO and DE algorithms. Then, Table 3 presents the results and comparison of MD-GWONM, MP-DE and other optimization algorithms. These algorithms are: Artificial Bee colony (ABC), Particle Swarm Optimization (PSO), Levy Flight Firefly Algorithm (LFFA), Gravitional Search Algorithm (GSA), Cuckoo Search Algorithm (CSA).

Table 1. Single-Objective Benchmark Test Functions

F.	Name	Test	D	LB	UB	Opt
f_1	Sumsquare	$f_1(x) = \sum_{i=1}^{N} x_i^4$	30	-10	10	0
f_2	Rastrigin	$f_2(x) = \sum_{i=1}^{N}\{x_i^2 - 10\cos(2\pi x_i) + 10\}$	30	-5.12	5.12	0
f_3	Quartic	$f_3(x) = \sum_{i=1}^{n} i x_i^4 + random(0,1)$	30	-1.28	1.28	0
f_4	Ackley	$f_4(x) = -20e^{-0.02\sqrt{D^{-1}\sum_{i=1}^{D} x_1^2}} - e^{D^{-1}\sum_{i=1}^{D}\cos(2\pi x_i)} + 20 + e$	30	-32	32	0
f_5	Alpine No.1	$f_5(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	30	-10	10	0
f_6	Griewank	$f_6(x) = \sum_{i=1}^{Dim} \frac{y_i^2}{4000} - \prod_{i=1}^{Dim} \cos\left(\frac{y_i}{\sqrt{i}}\right) + 1$	30	-600	600	0
f_7	Penalized	$f_7(x) = \sum_{i=1}^{Dim-1}(y_i - 1)^2 \times (1 + sin^2)(3\pi y_{i+1}) + (y_{Dim} - 1)^2(1 + sin^2(2\pi y_{Dim})) + sin^2(3\pi y_1)$	30	-50	50	0
f_8	Zakharov	$f_8(x) = \sum_{i=1}^{n} x_i^2 + (\frac{1}{2}\sum_{i=1}^{n} i x_i)^2 + (\frac{1}{2}\sum_{i=1}^{n} i x_i)^4$	30	-5	10	0
f_9	Sphere	$f_9(x) = \sum_{i=1}^{N} x_1^2$	30	-100	100	0

Table 2. Results of MDE-EGWO, DE, and GWO

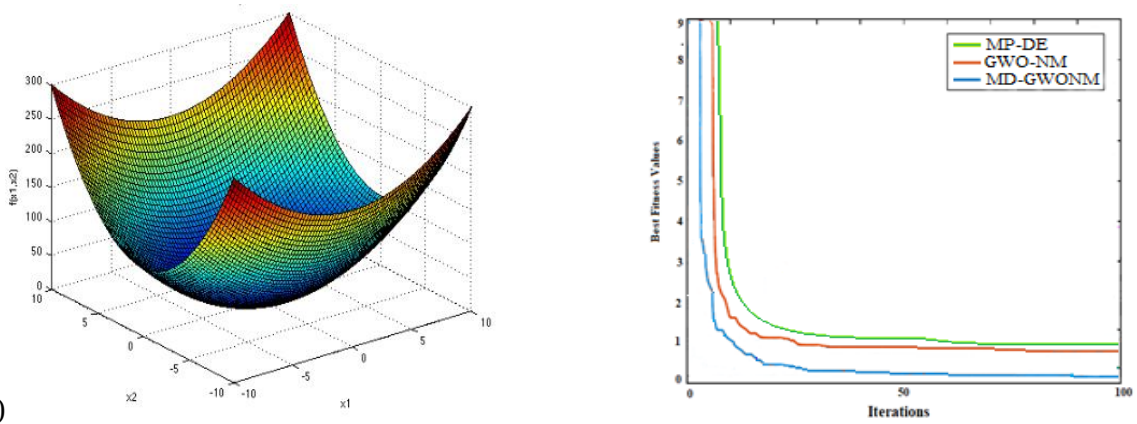
F	Statistics	DE	GWO	MDE-GWONM
f_1	Best	4.1458	3.34E-04	**0.00000**
	Mean	5.9475	0.00348	**0.00000**
	Std. Div	2.1354	3.12E-03	**0.00000**
f_2	Best	2.1049	0.00845	**0.00000**
	Mean	3.3085	0.08394	**0.00000**
	Std. Div	3.5478	0.01945	**0.00000**
f_3	Best	3.45892	0.02348	**0.00697**
	Mean	5.48953	0.03154	**0.01999**
	Std. Div	0.83211	0.00284	**0.00129**
f_4	Best	0.3247	0.4673	**0.00000**
	Mean	1.7798	1.6432	**0.00000**
	Std. Div	1.0422	1.0331	**0.00000**
f_5	Best	0.00064	0.00481	**0.00000**
	Mean	1.06309	0.08741	**0.00000**
	Std. Div	1.79308	0.03847	**0.00000**
f_6	Best	**0.00000**	0.001584	**0.00000**
	Mean	**0.00000**	0.009612	**0.00000**
	Std. Div	**0.00000**	0.084123	**0.00000**
f_7	Best	0.89765	0.12245	**0.00000**
	Mean	0.56432	0.26640	**0.00000**
	Std. Div	0.00318	0.05789	**0.00000**
f_8	Best	4112.205	3.59778	**0.00000**
	Mean	267.3249	3.94558	**0.00000**
	Std. Div	189.7456	0.87565	**0.00000**
f_9	Best	2.12461	0.00094	**0.00000**
	Mean	3.98452	0.00845	**0.00000**
	Std. Div	2.64871	0.05491	**0.00000**

Table 3. Results of MDE-GWONM and other optimization algorithm

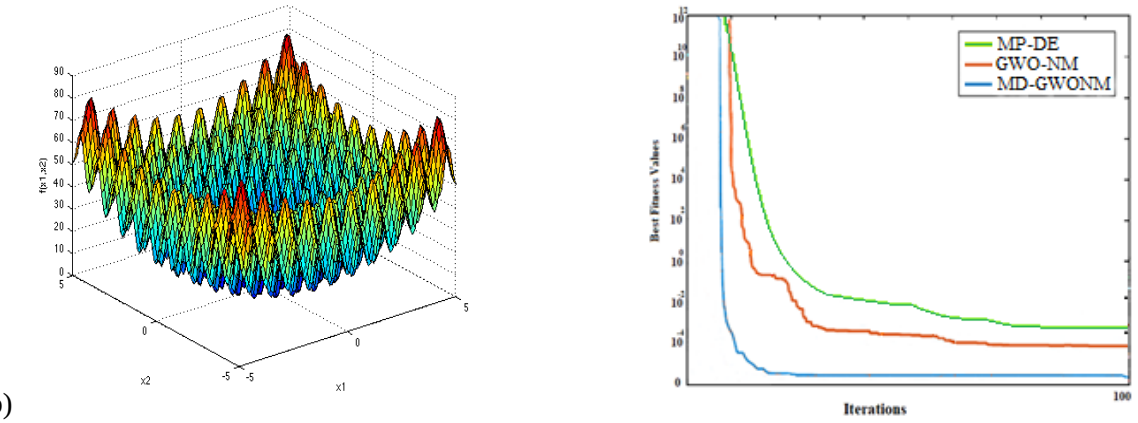
Fun	Statistics	ABC	PSO	LFFA	GSA	CSA	MDE-GWONM
f_1	Best	2.79E-16	2.13485	0.00774	0.00156	4.97E-04	**0.00000**
	Mean	2.72E-16	4.98451	0.21006	0.02943	0.00105	**0.00000**
	Std. Div	8.51E-12	3.94512	0.34752	0.08790	4.41E-04	**0.00000**
f_2	Best	1.40E-11	0.02448	4.94E-10	0.90001	**0.00000**	**0.00000**
	Mean	8.83E-13	2.15168	2.06E-07	1.00043	**0.00000**	**0.00000**
	Std. Div	2.76E-12	1.07664	5.18E-08	0.90536	**0.00000**	**0.00000**
f_3	Best	0.11531	1.3389	0.00409	0.06348	0.01741	0.00697
	Mean	0.19593	6.9606	0.02542	0.08815	0.02845	0.01999
	Std. Div	0.05549	0.6477	0.02312	0.04413	0.00148	0.00129
f_4	Best	2.8987	1.06481	1.598E-04	0.134432	**0.00000**	**0.00000**
	Mean	3.3345	2.92674	0.0014	0.765568	**0.00000**	**0.00000**
	Std. Div	0.6042	0.87441	5.409E-04	0.556324	**0.00000**	**0.00000**
f_5	Best	0.00042	0.00425	0.00024	0.00493	5.82E-05	**0.00000**
	Mean	0.28568	**2.67570**	0.00029	0.02171	2.48E-03	**0.00000**
	Std. Div	0.62473	12.3490	0.00037	0.00928	0.00048	**0.00000**
f_6	Best	4.261E-06	0.15676	3.20E-07	**0.00000**	0.00019	**0.00000**
	Mean	0.0035	0.24208	1.51E-06	**0.00000**	0.00048	**0.00000**
	Std. Div	0.0067	0.09374	1.88E-06	**0.00000**	0.00082	**0.00000**
f_7	Best	0.47989	5.523E+08	**0.00000**	15.3769	0.14548	**0.00000**
	Mean	0.44998	7.899E+08	**0.00000**	32366.20	1.16473	**0.00000**
	Std. Div	0.00478	1.439E+08	**0.00000**	59623.51	0.40721	**0.00000**
f_8	Best	7726.247	3.55412	4021.309	4214.467	3.55676	**0.00000**
	Mean	8094.705	4.77746	277.7689	345.7899	4.78767	**0.00000**
	Std. Div	246.1136	0.85447	171.7327	189.7867	0.89787	**0.00000**
f_9	Best	0.00432	1.2945	0.00128	0.04871	0.57843	**0.00000**
	Mean	0.00645	2.7707	0.00300	0.06643	0.76741	**0.00000**
	Std. Div	0.03184	1.0831	0.00105	0.00384	0.68817	**0.00000**

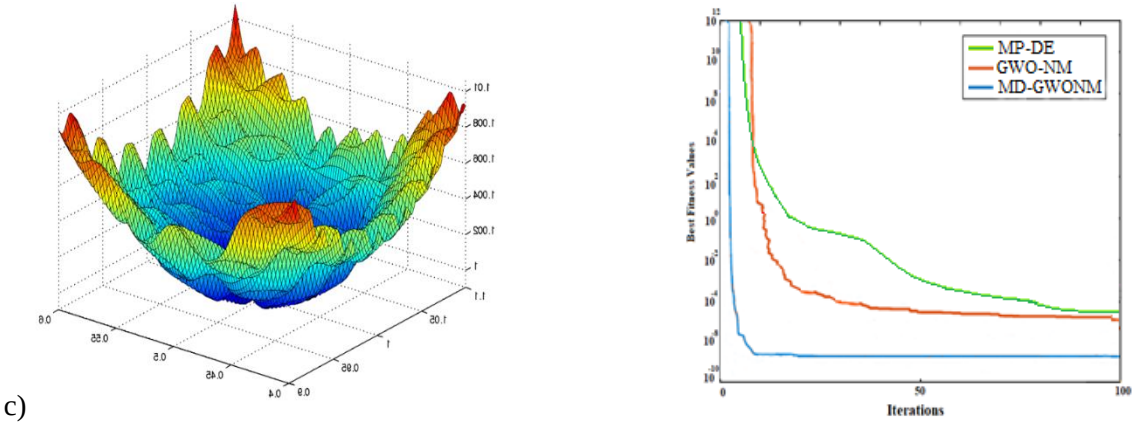
The convergence curves for several test functions of all proposed approaches are illustrated in following figures. Each figure, contains two parts, first, the three-dimensional illustrations of the test function, while second part presents the convergence of the algorithms. Figures below present the convergence for the first three test functions.

a)



b)





c)

Figure 3. The 3d plote and convergence analys for the first three test functions
a) f_1 b) f_2 c) f_3

5. Conclusion

Optimization problems is a type of problems when there is no linear method has the ability to handle them. In this paper, a new hybrid algorithm based on multi-population differential evolution and enhanced grey wolf optimizer based Nelder-Mead Method was proposed, which was called "MDE-GWONM". The proposed algorithm was used for handling global optimization test function. The results showed that our proposed algorithm have attained very good results as compared to the original versions of GWO and DE. In addition, it has showed a superior performance as compared to several state of arts optimization algorithms. For future studies, MDE-GWONM could be used for different optimization problems, such as training artificial neural network, or selecting the most relevant subset of features (i.e., feature selection problem).

References

[1] S. Koziel and X. S. Yang, *Computational Optimization, Methods and Algorithms*, vol. 356. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[2] T. Weise, M. Zapf, R. Chiong, and A. J. Nebro, "Why is optimization difficult?," in *Nature-Inspired Algorithms for Optimisation*, R. Chiong, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–50.

[3] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowledge-Based Syst.*, vol. 96, pp. 120–133, 2016.

[4] K. Hussain, M. N. Mohd Salleh, S. Cheng, and Y. Shi, "Metaheuristic research: a comprehensive survey," *Artif. Intell. Rev.*, vol. 52, no. 4, pp. 2191–2233, Dec. 2019.

[5] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah, *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*. Elsevier, 2018.

[6] A. Mortazavi, V. Toğan, and A. Nuhoğlu, "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems," *Eng. Appl. Artif. Intell.*, 2018.

[7] K. Sörensen, M. Sevaux, and F. Glover, "A history of metaheuristics," *Handb. heuristics*, pp. 1–18, 2018.

[8] M. Gendreau and J.-Y. Potvin, *Handbook of Metaheuristics*, vol. 57. Boston, MA: Springer US, 2003.

[9] X.-S. Yang and M. Karamanoglu, "Swarm Intelligence and Bio-Inspired Computation," in *Swarm Intelligence and Bio-Inspired Computation*, Elsevier, 2013, pp. 3–23.

[10] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, 1995, pp. 39–

43.

[11] X. S. Yang, "Firefly algorithms for multimodal optimization," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5792 LNCS, pp. 169–178, 2009.

[12] X. S. Yang, "A new metaheuristic bat-inspired algorithm," in *Studies in Computational Intelligence*, 2010, vol. 284, pp. 65–74.

[13] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theor. Comput. Sci.*, 2005.

[14] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *Int. J. Math. Model. Numer. Optim.*, 2010.

[15] S. Q. Salih and A. A. Alsewari, "A new algorithm for normal and large-scale optimization problems: Nomadic People Optimizer," *Neural Comput. Appl.*, vol. 32, no. 14, pp. 10359–10386, Jul. 2020.

[16] M. Kumar, A. J. Kulkarni, and S. C. Satapathy, "Socio evolution & learning optimization algorithm: A socio-inspired optimization methodology," *Futur. Gener. Comput. Syst.*, vol. 81, pp. 252–272, Apr. 2018.

[17] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, 2016.

[18] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems," *Comput. Des.*, vol. 43, no. 3, pp. 303–315, 2011.

[19] S. Mirjalili, "The ant lion optimizer," *Adv. Eng. Softw.*, vol. 83, pp. 80–98, 2015.

[20] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *J. Glob. Optim.*, vol. 11, pp. 341–359, 1997.

[21] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, 2014.

[22] S. Q. Salih, A. A. Alsewari, B. Al-Khateeb, and M. F. Zolkipli, "Novel Multi-Swarm Approach for Balancing Exploration and Exploitation in Particle Swarm Optimization," in *In Proceesdings of 3rd International Conference of Reliable Information and Communication Technology 2018 (IRICT 2018)*, 2018, pp. 196–206.

[23] A. H. Gandomi, X. S. Yang, S. Talatahari, and A. H. Alavi, "Firefly algorithm with chaos," *Commun. Nonlinear Sci. Numer. Simul.*, 2013.

[24] S. Gupta and K. Deep, "A novel Random Walk Grey Wolf Optimizer," *Swarm Evol. Comput.*, vol. 44, pp. 101–112, Feb. 2019.

[25] A. Ouaarab, B. Ahiod, and X. S. Yang, "Discrete cuckoo search algorithm for the travelling salesman problem," *Neural Comput. Appl.*, 2014.

[26] S. Q. Salih and A. A. Alsewari, "Solving large-scale problems using multi-swarm particle swarm approach," *Int. J. Eng. Technol.*, vol. 7, no. 3, pp. 1725–1729, 2018.

[27] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Inf. Sci. (Ny).*, 2013.

[28] H. A. Abdulwahab, A. Noraziah, A. A. Alsewari, and S. Q. Salih, "An Enhanced Version of Black Hole Algorithm via Levy Flight for Optimization and Data Clustering Problems," *IEEE Access*, vol. 7, 2019.

[29] S. Q. Salih, "A New Training Method Based on Black Hole Algorithm for Convolutional Neural Network," *J. Sourthwest Jiaotong Univ.*, vol. 54, no. 3, pp. 1–10, 2019.

[30] S. Gu, R. Cheng, and Y. Jin, "Feature selection for high-dimensional classification using a competitive swarm optimizer," *Soft Comput.*, vol. 22, no. 6, pp. 811–822, 2018.

[31] S. Ghimire, R. C. Deo, N. J. Downs, and N. Raj, "Self-adaptive differential evolutionary extreme learning machines for long-term solar radiation prediction with remotely-sensed MODIS satellite and Reanalysis atmospheric products in solar-rich cities," *Remote Sens. Environ.*, vol. 212, no. April, pp. 176–198, 2018.

[32] H. A. Ahmed, M. F. Zolkipli, and M. Ahmad, "A novel efficient substitution-box design based on firefly algorithm and discrete chaotic map," *Neural Computing and Applications*, 2018.

[33] A. A. Alzaidi, M. Ahmad, H. S. Ahmed, and E. Al Solami, "Sine-Cosine Optimization-Based Bijective Substitution-Boxes Construction Using Enhanced Dynamics of Chaotic Map," *Complexity*, vol. 2018, 2018.

[34] T. Hai, A. Sharafati, A. Mohammed, S. Q. Salih, R. C. Deo, N. Al-Ansari, and Z. M. Yaseen, "Global Solar Radiation Estimation and Climatic Variability Analysis Using Extreme Learning Machine Based Predictive Model," *IEEE Access*, vol. 8, pp. 12026–12042, 2020.

[35] H. A. Afan, M. F. Allawi, A. El-Shafie, Z. M. Yaseen, A. N. Ahmed, M. A. Malek, S. B. Koting, S. Q. Salih, W. H. M. W. Mohtar, S. H. Lai, A. Sefelnasr, M. Sherif, and A. El-Shafie, "Input attributes optimization using the feasibility of genetic nature inspired algorithm: Application of river flow forecasting," *Sci. Rep.*, vol. 10, no. 1, p. 4684, Dec. 2020.

[36] L. Penghui, A. A. Ewees, B. H. Beyaztas, C. Qi, S. Q. Salih, N. Al-Ansari, S. K. Bhagat, Z. M. Yaseen, and V. P. Singh, "Metaheuristic Optimization Algorithms Hybridized With Artificial Intelligence Model for Soil Temperature Prediction: Novel Model," *IEEE Access*, vol. 8, pp. 51884–51904, 2020.

[37] O. Kisi, S. Heddam, and Z. M. Yaseen, "The implementation of univariable scheme-based air temperature for solar radiation prediction: New development of dynamic evolving neural-fuzzy inference system model," *Appl. Energy*, 2019.

[38] S. Q. Salih, A. A. Alsewari, and Z. M. Yaseen, "Pressure Vessel Design Simulation: Implementing of Multi-Swarm Particle Swarm Optimization," *Proc. 2019 8th Int. Conf. Softw. Comput. Appl.*, pp. 120–124, 2019.

[39] S. Das and P. N. Suganthan, "Differential Evolution: A Survey of the State-of-the-Art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[40] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution – An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

[41] F. Neri and V. Tirronen, "Recent advances in differential evolution: A survey and experimental analysis," *Artif. Intell. Rev.*, vol. 33, no. 1–2, pp. 61–106, 2010.

[42] G. Wu, X. Shen, H. Li, H. Chen, A. Lin, and P. N. Suganthan, "Ensemble of differential evolution variants," *Inf. Sci. (Ny).*, 2018.