Enhanced Jaya Algorithm for Multi-objective Optimisation Problems

Rahaini Mohd Said[1], Roselina Sallehuddin[2], Nor Haizan Mohd Radzi[3], Wan Fahmn Faiz Wan Ali[4]
Faculty of Computing, Universiti Teknologi Malaysia[1]
Faculty of Electrical and Electronic Engineering Technology, Universiti Teknikal Malaysia Melaka[1]
Faculty of Computing, Universiti Teknologi Malaysia[2, 3]
Faculty of Mechanical Engineering, Universiti Teknologi Malaysia[4]

***Abstract*—Evolutionary algorithms are suitable techniques for solving complex problems. Many improvements have been made on the original structure algorithm in order to obtain more desirable solutions. The current study intends to enhance multi-objective performance with benchmark optimisation problems by incorporating the chaotic inertia weight into the current multi-objective Jaya (MOJaya) algorithm. Essentially, Jaya is a recently established population-oriented algorithm. Exploitation proves to be more dominant in MOJaya following its propensity to capture local optima. This research addressed the aforementioned shortcoming by refining the MOJaya algorithm solution to update the equation for exploration-exploitation balance, enhancing divergence, and deterring premature convergence to retain the algorithm fundamentals while simultaneously sustaining its parameter-free component. The recommended chaotic inertia weight-multi-objective Jaya (MOiJaya) algorithm was assessed using well-known ZDT benchmark functions with 30 variables, whereas the convergence matrix (CM) and divergence matrix (DM) analysed the suggested MOiJaya algorithm performances are inspected. As such, this algorithm enhanced the exploration-exploitation balance and substantially prevented premature convergence. Then, the proposed algorithm is compared with a few other algorithms. Based on the comparison, the convergence metric and diversity metric results show that the recommended MOiJaya algorithm potentially resolved multi-objective optimisation problems better than the other algorithms.**

Keywords*—*MOJaya; chaotic inertia weight; ZDT benchmark function; convergence metric; diversity metric

I. Introduction

As a mathematical instrument that adequately models and solves real-life complexities with multiple objectives and simultaneous enhancement, multi-objective optimisation and its relevant intricacies have garnered much scholarly attention across various disciplines, specifically engineering and sciences. Multiple population-oriented metaheuristic algorithms were recommended for multi-objective problem (MOP) solving. As MOP goals must simultaneously optimise the conflicting nature of multiple objectives given the absence of one distinct alternative to optimise all collaborative counterparts [1], a set of optimal trade-off alternatives (Pareto) was employed as a solution. Thus, a single and optimal solution is non-existent in this regard. Evolutionary algorithms or EAs imply some of the most extensively-utilised algorithms to solve MOPs and numerous issues with competing objectives across industrial, engineering, and research disciplines [2]. Robust optimisation algorithms serve to resolve intricate real-world MOPs in one run. Relevant research has demonstrated the successful application of multi-objective optimisation, which has extended EAs entailing MOPSO [3], [4], MODE [5], MOACO [6], and MOGA [7] for improved performance in multi-objective optimisation problem solution.

Recent EA developments in the past few decades have rendered the algorithms an efficient means of solving intricate multi-objective evolutionary algorithms (MOEAs). Their competence in simultaneously examining different regions of the Pareto front (PF) and generating a set of Pareto solutions in one run facilitates scholars toward multi-objective and real-life optimisation problems using distinct domains. Nevertheless, the algorithms types rely on algorithm-centric parameter refinements with common controlling parameters. Such specific parameters entail multiple purposes and impact the convergence rates, diversity, efficiency, scalability, exploration, and exploitation within the solution. For example, MOGA constitutes a mutation and crossover operator to attain the exploration and exploitation mechanism. Rao and colleagues established the parameter-free Jaya and MOJaya algorithms in 2016 to address scholars' complexities and control specific parameters for algorithm simulation.

The aforementioned algorithms prove adequate in engineering areas, which entail i) multiple variables and parameters that require observation and ii) control or uncontrol parameters that are deemed challenging to manage without expertise and experience. Specifically, the Jaya algorithm or EA optimisation, which only requires several turning control parameters (population size, number of generations, and design variables), could only alleviate specific complexities. The EAs optimisation has recently solved applications for a single objective, whereas MOJaya resolved multi-objective optimisation problems. The solution consistently shifts towards the most and least optimal solutions that are avoided in simulation under the Jaya algorithm concept. The Update phase serves to modify the solution from earlier generations compared to other algorithms, which require two phases to refine solutions involving teaching-learning optimisation (TLBO), artificial bee colony (ABC), and differential evolution (DE). Intriguingly, the Jaya algorithm optimally manages continuous, discrete, and integer variables [8].

Hence, in this study, the improved MOJaya algorithm is proposed to solve multi-objective optimisation problems, while the properties of MOJaya are preserved. The non-dominated

sorting (NDS) approach with a reference point is used to find dominance relation, where NDS is performed according to the Euclidean distances between each possible solution of the front and reference point. The solution update equation is modified by incorporating the chaotic inertia weight. Finally, the recommended algorithm's performance were tested using ZDT benchmark test problems. The convergence metric and diversity metric evaluate the performance of the recommended algorithms.

The remaining paper is organized as follows: Section II, discusses the improvement Jaya Algorithm and its variations by the previous researcher. Then, the recommended algorithm methodology is presented in Section III. Next, Section IV presents the obtained results and analysis. Finally, the conclusion and future research are presented in Section V.

II. MOJAYA ALGORITHM

Rao's [9] Jaya algorithm is a novel, astute, population-oriented, and parameter-free solution that manages constrained and unconstrained optimisation-related problems. Resultantly, the number of function assessments required to obtain a solution is lesser than that of TLBO. Several limitations have been ascertained despite its consistent attempts to omit the least optimal solutions and iterate the most optimal solution search space. For example, the Jaya algorithm becomes trapped in local optimal solutions where exploitation overrides exploration [10]. The basic Jaya algorithm is updated to relieve researchers (specifically in complex engineering fields) from refining algorithm-centric parameters.

Following past literature, the Jaya algorithm functions to solve multiple real-time and standard benchmark functions with distinct components and variants. This section elaborates on several parameter-free Jaya algorithm variants that were published in relevant research. The MOJaya, a posteriori version of Jaya, was developed in 2017 to solve multi-objective optimisation problems. Although MOJaya algorithm solutions are similarly updated to that of Jaya, MO-Jaya incorporates non-dominated sorting and the crowding distance computing mechanism for successful multi-objective management.

Past scholars demonstrated inconsistent outcomes in terms of multi-objective Jaya algorithm enhancement. Rao and Pawar (2020) recommended a novel and upgraded version (Rao's quasi-oppositional approach) under the Jaya algorithm with multi-objective and quasi-oppositional-oriented-learning techniques to address the diversity in the algorithm searching process. Three multi-objective optimisation case studies involving real-world and intricate engineering optimisation problems were applied with a single-layered microchannel heat sink (SL-MCHS), a double-layered microchannel heat sink (DL-MCHS), and a plate-fin heat sink (PFHS) to examine the recommended algorithm efficiency. The findings derived through the recommended algorithms were compared against those elicited with advanced optimisation algorithms: GA, ABC, DE, PSO, TLBO, MOGA, NSGA-II, real-coded GA (RCGA), direction-based GA, and basic and self-adaptive multi-population (SAMP) Rao algorithms. Essentially, the recommended algorithms proved superior and competitive compared to other optimisation counterparts [11].

The discrete multi-objective Jaya algorithm potentially addresses the flexible job-shop scheduling problem (FJSSP) by regarding the minimisation of makespan and total and critical machine workload as performance measures. Dynamic mutation operator and modified crowding distance measures were proposed to improve search process diversity. In-depth computational experiments were performed by regarding 203 FJSSP instances from past research. A comparison between the recommended algorithm and the weighted sum version demonstrated higher performance than the other approach and other MOEAs. Based on the computational outcomes, the proposed algorithm efficiently obtained diverse and enhanced Pareto-optimal solutions [12].

Rao and Hameer (2019) presented an adaptive multi-team perturbation with multiple teams to navigate the Jaya algorithm and examine its search space. The recommended algorithm was investigated with two multi-objective optimisation case studies of a solar dish Stirling heat engine system and one counterpart of the Stirling heat pump. The suggested algorithm utilised various perturbation equations with dominance principles and the crowding distance estimation approach for simultaneous multiple objective management. As a decision-making approach, the 'technique for order of preference by similarity to ideal solution' was also utilised for optimal solution identification. The computational outcomes elicited by the suggested algorithm proved superior to those attained by other study algorithms [13].

Rao and Saroj (2018) recommended a novel and unique Jaya algorithm for multi-objective design optimisation problem solution. The aforementioned algorithm was applied to resolve the heat exchanger design problem where two conflicting objectives (optimise heat exchangers' total annual cost and effectiveness) were simultaneously regarded. The least optimal Jaya algorithm solutions were substituted with an elitist value at the end of each iteration. As such, local trapping was prevented by arbitrarily selecting duplicate solution parameters. The recommended algorithm also optimally solved the multi-objective heat exchanger design compared to GA and TLBO [14].

The drawbacks inherent in the basic Jaya algorithm were highlighted in past studies. The exploration-exploitation balance implies one of the success criteria for any nature-oriented algorithm. Exploitation depicted a more significant impact than exploration as this algorithm catalyses objective function value towards the most optimal solution space. In this vein, local minima are trapped with less diverse solutions. The Jaya algorithm, which causes premature convergence and impacts solution quality, only upgrades the solution with the most and least optimal solutions from past iterations. Another shortcoming denotes the basic Jaya algorithm weakness regarding information exchange among individuals with no local minima mechanism in the event of a trapped algorithm. Palpably, relevant scholars have strived to integrate multiple methods for the optimal performance of balancing exploration and exploitation capacities and solving multiple real-time and benchmark problems. Exploration-exploitation balance is a pivotal mechanism that efficiently assesses the optimisation algorithm [15]–[17]. The numerous Jaya algorithm refinements

presented in this study primarily emphasised mitigating the basic Jaya algorithm limitations.

However, the results produced inconsistent results while comparing the performance with other results. Hence, the current research strived to (i) employ and assess the proposed techniques for improved MOJaya algorithm performance, (ii) solve multi-objective optimisation problems, and (iii) introduce a chaotic sequence. Inertia chaotic weight was initially adopted in the solution for the updated equation to be elicited from the local optimal solution. Regarding MOJaya, the most optimal solution was derived from the sorted list through non-dominant sorting and a crowding distance mechanism to eliminate the least optimal counterparts, thus leading to local region searches, premature convergence, and lesser diverse solutions.

The previous study observed that researchers are working to improve the MOJaya algorithm's performance**.** Various techniques are adopted to improve performance and also focus on real-life MOP. The motivation for this paper is as follows. First, the recommended study strategies for the algorithm is presented to improve the exploration of the existing MOJaya algorithm. The recommended algorithm development towards high performance while balancing exploration and exploitation is adopting inertia chaotic weight in the solution for an updated equation provides the best solution for promising and sparse search space regions. Then, the recommended algorithm's performance is evaluated using a two-objective ZDT test performance. Finally, the exiting MOJaya and MOiJaya algorithm with other multi-objective optimization algorithm performance are compared using the convergence and diversity metrics.

III. The MOIJAYA Algorithm

The MOJaya, a posteriori Jaya algorithm version, was employed for stochastically updated Multi-objective optimization problems (MOOP) solutions. The most optimal solution denoted maximum fitness, whereas the worst counterpart implied minimum fitness. The MOJaya algorithm was incorporated into a non-dominant ranking and crowding distance evaluation method to resolve multi-objective algorithm. The MOJaya algorithm pseudo-code is depicted in Algorithm 1.

A. Chaotic Random Inertia Weight Mechanism

The chaos mechanism, a well-established logistic map, was integrated with TLBO and the basic Jaya algorithm for single optimisation. Meanwhile, the chaotic sequence implied two attributes (ergodicity and randomness) that drove the algorithm away from the optimal local solution and enhanced solution quality [18], [19].The fundamentals of chaotic inertia weight aimed to establish the inertia weight coefficient with chaotic mapping, which is integrate Logistic mapping, the method used in this paper. Inertia weight denotes one of the PSO algorithm parameters that substantially affect global and local searches and PSO algorithm performance through simultaneous particle retention in inertial motion and search space expansion [20]. The chaotic random inertia weight generated as (1) and (2) was incorporated into this study, given its value in demonstrating particle velocity shifts.

Logistic mapping,

$$z = 4 \times z \times (1 - z) \tag{1}$$

Chaotic Inertia weight

$$\varpi = 0.5 \times random(\,) + 0.5 \times z \tag{2}$$

Algorithm 1 : MOJaya algorithm 1

Begin

Step 1 Initialize the algorithm's control parameters, such as population size (N) and the maximum number of iterations (Gmax). Also initialize the problem-specific parameters such as a number of objectives (M).

Step 2 Generate the initial solution randomly and evaluate it for each individual. Identify the best and worst solutions.

Step 3 Find the new solution for all the individuals in a population using the modified solution update equation.

Step 4 Evaluate the modified solution using Non-dominated sorting with a and ranking method solution using Non-dominated sorting with ranking method.

Step 5 Combine the new solution with the old one. Sort the combined solution in ascending order. Identify the best and worst from the selected solution to update the solution in the next iteration.

Step 6 Update the global best solution by comparing the old global best solution with the new best solution.

Step 7 If the number of iterations reaches the maximum, stop the procedure and report the global best solution; otherwise, repeat the procedure from Step 3.

End

B. Proposed Modification in MOiJaya

The MOJaya algorithm, which entails a non-dominated sorting scheme and crowding distance from the NSGA-II counterpart, is developed to solve the multi-objective optimisation problem by integrating components from current MOEAs. The MOJaya algorithm performs sorting in ascending order by integrating the solutions from present and past iterations. The best solutions for the first N (population size) were selected through the non-dominated ranking scheme from the sorted list. Meanwhile, the most and least optimal solutions were selected from the N (population size) to update the following iteration. The most optimal and random solutions were chosen as the next search direction to explore more space in the early iteration of the algorithm process. The most optimal solution candidate in the algorithm functions to navigate the population towards a better region, whereas a random solution facilitates search space expansion. Searching local region

outcomes results in premature convergence and low diversified solutions as this approach disregards the least optimal solution.

The recommended algorithm and solution update equation was refined by integrating the chaotic sequence to alleviate MOJaya limitations. The random number employed in the MOJaya solution for an updated equation was substituted with a chaotic random inertia weight. The chaotic random inertia weight attributes entailing ergodicity and randomness enabled the algorithm to drive away from the optimal local solution: a regulation to balance the stochastic search and local search probabilities. Additionally, the recommended algorithms were improved in terms of convergence metric and diversity metric. The refined solution for an updated equation is expressed in (3) below:

$$x_{(i+1,j,k)} = x_{(i,j,k)} + \varpi_{(i,j,1)} * \left[x_{(i,j,w)} - \left| x_{(i,j,k)} \right| \right] - \varpi_{(i,j,2)} * \left[x_{(i,j,w)} - \left| x_{(i,j,k)} \right| \right] \quad (3)$$

The pseudocode for MOiJaya is depicted in Algorithm 2. Fig. 1 illustrates the flowchart parallel to the previously-mentioned refinements.

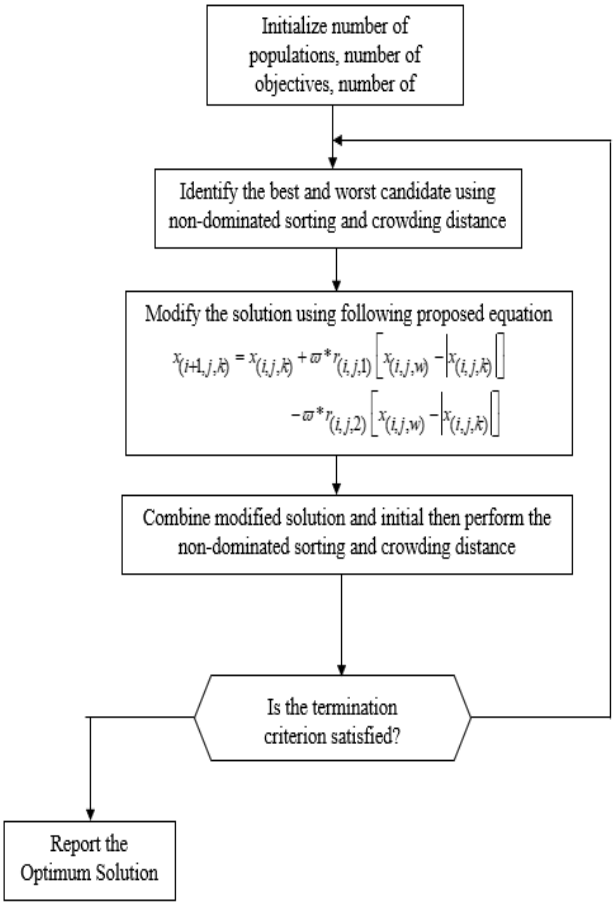

Fig. 1. The Flowchart of MOiJaya Algorithm.

Algorithm 2: MOiJaya algorithm

Begin
Input: Population size N, number of objectives, number of design variables, number of iterations, constrain, and function
Output: Solution f_1 and f_2
1: Initialize population size, initialize the best and worst
2: Generate initial population randomly
For i< var
Generate initial population for variable and objective
End
Find the best and worst candidates using non-dominated sorting and crowding distance
While t<iter.max
Generate new solution using modified solution update equation as equation 3
Evolve
Merge the new and old solution
Perform non-dominated sorting and crowding distance
Select the first N solution
t+1;
End
End

IV. Computational Results and Analysis

This section highlights the recommended MOiJaya algorithm assessment. The suggested chaotic-oriented MOJaya algorithm was incorporated into Matlab2020 and examined with ZDT1-ZDT6 for multi-objective benchmark functions using two objectives with 30 variables. The outcomes elicited by the MOiJaya algorithm were compared against those identified in past empirical works with well-known CM and DM. The ZDT functions in PF demonstrated diverse attributes (separable, multimodal, linear, concave, mixed, and biased) that rendered it challenging to solve the problems with MOEAs (Deb et al., 2005). Table I presents the ZDT1-ZDT6 objective test problems.

The suggested MOiJaya algorithm extends to a renowned basic Jaya algorithm that only requires common controlling parameters. The following common controlling parameters are utilised to perform experiments:

Population size : 100

Maximum iteration : 10000

The recommended chaotic-based MOJaya algorithm was compared against SPEA2, NSGA-II, and EDATCMO [21] with two popular MOEA performance metrics (DM and CM) to assess the convergence and diversity solution for Pareto-front.

A. Perormance Measures

The common trend in various successful solution methodology development, including MOEAs, involves performance comparison on multiple test problems. Two goals

(discover solutions with close proximity to the Pareto-optimal solutions and alternatives that are distinctly diverse in the elicited non-dominated front) were identified in multi-objective optimisation, unlike the single counterpart. A minimum of two performance metrics proves vital towards the actualization of both multi-objective optimisation goals to ascertain an effective MOEA (Deb, 2001). One performance matrix analyses the progress towards the Pareto-optimal front while the other assesses the solution spread. Three common metrics were employed in this study despite the numerous performance metrics highlighted in past literature, two of which would be implemented for MDEA analysis: CM and DM.

The CM was recommended by [22] to assess the distance between the obtained non-dominated (NF) front and optimal PF. The mathematical equations are expressed as follows (4):

$$d = \min_{j=1} N \sqrt{\sum_{k=1}^{M} \left(\frac{f_k(i) - f_k(j)}{f_k^{\max} - f_k^{\min}} \right)} \quad (4)$$

$$CM = \left(P(t)\right) = \frac{\sum_{i=1}^{|F(t)|} d_i}{|F(t)|} \quad (5)$$

Specifically, M denotes the number of objectives, $f_k^{\max} - f_k^{\min}$ imply the maximum and minimum function values of k^{th} and the objective function in the reference set, respectively, and N indicates the reference set size. The minimum value of CM, (5) reflects improved multi-objective optimisation algorithm performance.

The DM was suggested by [22] to assess the extent of solution spread, not unlike DM where optimal algorithms denote lower diversity metrics for evenly-spread solutions on PF. The spread was ascertained by computing each solution gap in PF with the neighbouring solution. The mathematical equation is expressed as follows:

$$DM = \frac{d_f + d_l + \sum_{i=1}^{N-1} d_i - \bar{d}}{d_f + d_l + \left((N-1)\right)\bar{d}} \quad (6)$$

Based on the Euclidean distance between the consecutive solutions within the obtained non-dominated set of solutions, $\bar{d}$ denotes the average of all distances, $d_i \left(i = 1, 2, ..., N\right)$ assumes the N solutions in the derived non-dominated set, and d_l reflects the Euclidean distance between extreme and boundary solutions. The $\left(N-1\right)$ is utilised as the solution distance that constitutes two solutions.

B. Results on Zdt Test Functions

In this section, five benchmarks are used to test the proposed algorithm and analyzed the obtained results. The five well-known benchmarks of optimization problems have been listed in Table I. The optimal values of all minimization objective function given in Table I is zero, which is, $f(x_1) = 0$ where x_1 is the optimal solution.

The recommended MOiJaya algorithm was performed 30 times for the chosen test function and corresponding number of objectives. In Table II-VI, we compared the proposed algorithm with the five others algorithm. The purpose of this test is to evaluate the quality of the solutions of purposed algorithm in different benchmarks compared to other algorithms. The mean and standard deviation (sd) values of DM and CM through SPEA2[22] ,NSGAII [23], FastPGA , and EDATCMO [21], MOJaya, and suggested MOiJaya algorithms on multi-objective benchmark functions ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6. The MOJaya algorithm employs the solution for an updated equation from the basic Jaya counterpart [24]. The common controlling parameters remain the same for both MOJaya and MOiJaya.

C. Results on Zdt Test Functions

In this section, five benchmarks are used to test the proposed algorithm and analyzed the obtained results. The five well-known benchmarks of optimization problems have been listed in Table I. The optimal values of all minimization objective function given in Table I is zero, which is, $f(x_1) = 0$ where x_1 is the optimal solution.

The recommended MOiJaya algorithm was performed 30 times for the chosen test function and corresponding number of objectives. In Table II-VI, we compared the proposed algorithm with the five others algorithm. The purposed of this test is to evaluate the quality of the solutions of purposed algorithm in different benchmarks compared to other algorithms. The mean and standard deviation (sd) values of DM and CM through SPEA2[22] ,NSGAII [23], FastPGA , and EDATCMO [21], MOJaya, and suggested MOiJaya algorithms on multi-objective benchmark functions ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6. The MOJaya algorithm employs the solution for an updated equation from the basic Jaya counterpart [24]. The common controlling parameters remain the same for both MOJaya and MOiJaya.

In Fig. 2, Pareto optimal for MOiJaya are presented. It is found that the generated Pareto optimal front for MOiJaya quite close to True Pareto front. The convergence metric and divergence metric results for ZDT1 problems in Table II show that MOiJaya has lower values in the metric than all the other algorithms. This shows that MOiJaya is better in diversity and converge than other algorithms.

Fig. 3 shows the Pareto optimal fronts MOiJaya for test problems ZDT2. It is found that from MoiJaya is better in the uniform spread. Though MOiJaya pareto optimal slightly close to True pareto, MOiJaya is better diversity than MOJaya. Table III shows the results of convergence and divergence metric for ZDT2 problems. It is found that MOiJaya is better in diversity than all the other algorithms. From the results, the MOiJaya is worse in convergence metric than SPEA, NSGA-II, FastPGA and EDATCMO but better than MOJaya.

TABLE I. THE ZDT TEST PROBLEMS [22]

Problem	N	Variable bounds	Objective Functions	Optimal Solutions	Comments
ZDT 1	30	[0,1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{x_1 / g(x)}\right]$ $h(f_1) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)(n-1)$	$x_1 \in [0,1]$ $x_1 = 0$ $i = 2,\ldots,n$	Convex
ZDT 2	30	[0,1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - (x_1/g(x))^2\right]$ $h(f_1) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)(n-1)$	$x_1 \in [0,1]$ $x_1 = 0$ $i = 2,\ldots,n$	nonconvex
ZDT 3	30	[0,1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{x_1 / g(x)} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right]$ $h(f_1) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)(n-1)$	$x_1 \in [0,1]$ $x_1 = 0$ $i = 2,\ldots,n$	Convex Disconnected
ZDT 4	30	[0,1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{x_1 / g(x)}\right]$ $h(f_1) = 1 + 10(n-1) + \sum_{i=2}^{n} x_i^2 - 10\cos(4\pi x_i)$	$x_1 \in [0,1]$ $x_1 = 0$ $i = 2,\ldots,n$	Convex
ZDT 6	30	[0,1]	$f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(x) = g(x)\left[1 - (x_1/g(x))^2\right]$ $h(f_1) = 1 + 9\left[\left(\sum_{i=2}^{n} x_i\right)(n-1)\right]^{0.25}$	$x_1 \in [0,1]$ $x_1 = 0$ $i = 2,\ldots,n$	Nonconvex Nonuniformly spaced

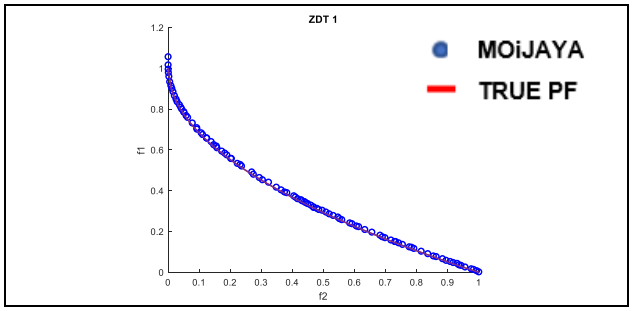

Fig. 2. The Results of Test Problem ZDT1.

TABLE II. COMPARISON OF CM AND DM VALUES OBTAINED WITH ALL ALGORITHMS FOR ZDT1 PROBLEMS

Case	Algorithm	Convergence (mean±sd)	Diversity (mean±sd)
ZDT1	**SPEA**	0.09462±0.04511	0.42209±0.01012
	NSGA-II	0.10872±0.00362	0.50827±0.02446
	FastPGA	0.09154±0.00621	0.70009±0.01174
	EDATCMO	0.02363±0.00146	0.21738±0.00368
	MOJAYA	0.23346±0.05597	0.02044±0.04281
	MOiJAYA	**0.02226±0.00287**	**0.00168±0.00232**

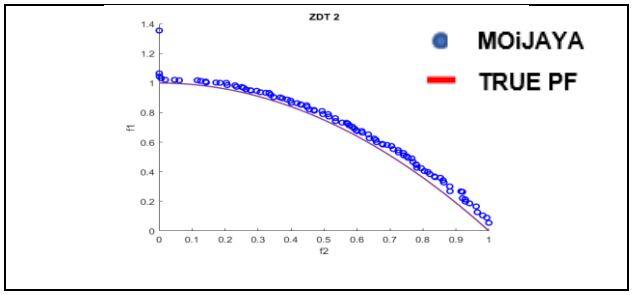

Fig. 3. The Results of Test Problem ZDT2 MOiJaya.

TABLE III. COMPARISON OF CM AND DM VALUES OBTAINED WITH ALL ALGORITHMS ZDT2 PROBLEMS

Case	Algorithm	Convergence	Diversity
		(mean±sd)	(mean±sd)
ZDT2	**SPEA**	0.08073±0.06101	0.50013±0.01612
	NSGA-II	0.09023±0.00401	0.30163±0.01503
	FastPGA	0.02067±0.00702	0.60034±0.2984
	EDATCMO	**0.00821±0.00472**	0.20022±0.00863
	MOJAYA	0.26564±0.22560	0.04489±0.02727
	MOiJAYA	0.15267±0.27201	**0.03300±0.02923**

From the Fig. 4, it is found that the generated Pareto optimal front for MOiJaya quite close to True Pareto front. This is also confirmed with the results of convergence and diversity in Table IV. Comparison with other algorithms shows that MOiJaya outperforms all of them in both metrices.

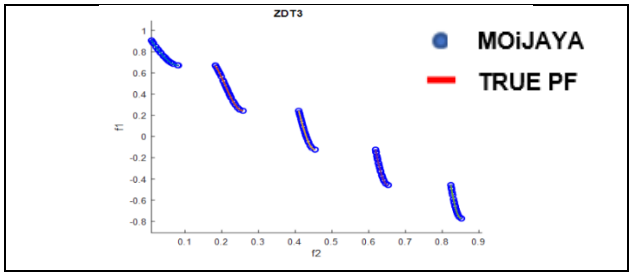

Fig. 4. The Results of Test Problem MOiJaya.

TABLE IV. COMPARISON OF CM AND DM VALUES OBTAINED WITH ALL ALGORITHMS ZDT3 PROBLEMS

Case	Algorithm	Convergence	Diversity
		(mean±sd)	(mean±sd)
ZDT3	**SPEA**	0.13996±0.07603	0.74932±0.05006
	NSGA-II	0.04208±0.09104	0.85061±0.09025
	FastPGA	0.07024±0.06541	0.85061±0.06453
	EDATCMO	0.02005±0.01843	0.50064±0.06132
	MOJAYA	0.04657±0.00559	0.06691±0.12251
	MOiJAYA	**0.03764±0.05113**	**0.02896±0.00584**

It is found in Fig. 5 that MOiJaya is far from True Pareto for this problem. Table V shows the results of convergence and divergence metrics. It is found that MOJaya is better in convergence than all the other algorithm but very close to MOiJaya. But, from the results of diversity, MOiJaya is better than all the other algorithms.

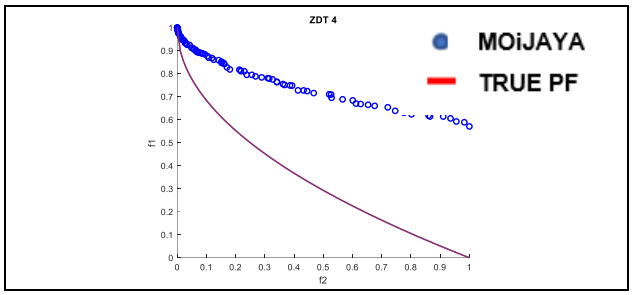

Fig. 5. The Results of Test Problem ZDT4 MOiJaya.

TABLE V. COMPARISON OF CM AND DM VALUES OBTAINED WITH ALL ALGORITHMS ZDT4 PROBLEMS

Case	Algorithm	Convergence	Diversity
		(mean±sd)	(mean±sd)
ZDT4	**SPEA**	0.62768±0.10676	0.51682±0.07319
	NSGA-II	1.13458±0.90054	0.85454±0.09003
	FastPGA	2.9742±1.94585	0.96856±0.07032
	EDATCMO	0.50547±0.08019	0.96856±0.07032
	MOJAYA	**0.21701±0.05162**	0.0255±0.06448
	MOiJAYA	0.27136±0.11211	**0.01204±0.00448**

Test problems ZDT6 Pareto optimal front is presented in Fig. 6, It is found that the generated Pareto optimal front for MOiJaya quite close to True Pareto front. Results in Table VI shows that MOiJaya outperforms all other algorithms in test problem ZDT6 in both metrices.

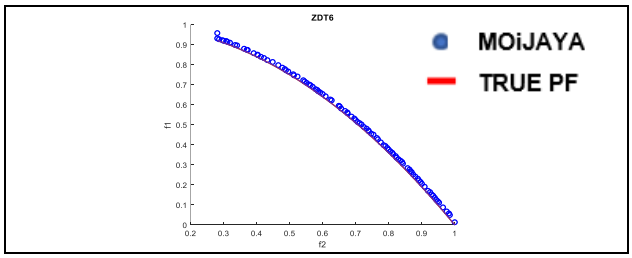

Fig. 6. The Results of Test Problem ZDT6 MOiJaya.

TABLE VI. COMPARISON OF CM AND DM VALUES OBTAINED WITH ALL ALGORITHMS ZDT6 PROBLEMS

Case	Algorithm	Convergence	Diversity
		(mean±sd)	(mean±sd)
ZDT6	**SPEA**	0.91036±0.09105	0.74532±0.04093
	NSGA-II	0.49311±0.00874	0.63847±0.08006
	FastPGA	0.72347±0.01106	0.84442±0.04034
	EDATCMO	0.10048±0.01002	0.31618±0.01782
	MOJAYA	0.78092±0.47053	0.07634±0.08416
	MOiJAYA	**0.07094±0.11630**	**0.02013±0.02014**

In Fig. 2-6, MOiJaya has achieved the two goals of multi-objective optimization: convergence to the true Pareto-front (ZDT 1 and ZDT3) and uniform spread of solution along the front (ZDT1-ZDT6). In all the Tables II-VI, MOiJaya demonstrated better "optimal" values for CM involving ZDT1, ZDT3, and ZDT6 and DM involving all benchmark functions: ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6. Additionally, the results are achieved the following points; in basic MOJaya algorithm, the exploitation is dominate, the recommended algorithm was modified to balance the exploration and exploitation by reducing the dominance of exploitation behavior. Other, using the chaotic inertia weight improve the converge rate and balance the exploration and exploitation. Specifically, the MOiJaya outcomes proved comparable to MOJaya. The MOiJaya curves in all the aforementioned figures were comparable to that of MOJaya regarding convergence and diversity. Conclusively, the performance of MOiJaya in solving multi-objective optimisation problems is deemed encouraging and comparable to other MOEAs.

V. Conclusion

The current study introduced an enhanced MOiJaya algorithm to solve multi-objective optimisation problems. Specifically, the proposed method enhanced the current MOJaya algorithm through the chaotic inertia weight-oriented logistic chaotic sequence. The chaotic inertia weight was incorporated into the solution for an updated equation of the current MOJaya to mitigate the dominance of exploitation in this algorithm. The recommended modification also enhanced the multi-objective Jaya algorithm searchability. Chaotic inertia weight was integrated with MOJaya to improve exploration and the exploitation-exploration balance. The proposed MOiJaya approach efficiency was analysed with a benchmark performance ZDT test that was assessed with CM and DM. The recommended algorithm outcomes were compared to the most established findings following past research. Post-comparison, MOiJaya outperformed MOJaya with relatively good performance compared to advanced empirical algorithms. Nevertheless, there is much room for improvement although the suggested MOiJaya algorithm optimised the exploration-exploitation balance. Further research could assess the recommended method with other multi-objective benchmark datasets and real-time multi-objective optimisation problems from various domains.

Acknowledgment

Special appreciation to reviewers for the useful advice and comments. This research was supported by Ministry of Higher Education (MOHE) through Fundamental Research Grant Scheme FRGS/1/2019/ICT02/UTM/02/13 and Research Management Centre (RMC), UTM.

References

[1] L. Fan, T. Yoshino, T. Xu, Y. Lin, and H. Liu, "A Novel Hybrid Algorithm for Solving Multiobjective Optimization Problems with Engineering Applications," Math. Probl. Eng., vol. 2018, 2018, doi: 10.1155/2018/5316379.

[2] J. A. Adeyemo and F. A. O. Otieno, "Multi-Objective Differential Evolution Algorithm for Solving Engineering Problems," J. Appl. Sci., vol. 9, no. 20, pp. 3652–3661, 2009, doi: 10.3923/jas.2009.3652.3661.

[3] G. C. M. Patel, P. Krishna, P. R. Vundavilli, and M. B. Parappagoudar, "Multi-Objective Optimization of Squeeze Casting Process using Genetic Algorithm and Particle Swarm Optimization," Arch. Foundry Eng., vol. 16, no. 3, pp. 172–186, 2016, doi: 10.1515/afe-2016-0073.

[4] G. C. Manjunath Patel, P. Krishna, and M. B. Parappagoudar, "Modelling and multi-objective optimisation of squeeze casting process using regression analysis and genetic algorithm," Aust. J. Mech. Eng., vol. 14, no. 3, pp. 182–198, 2016, doi: 10.1080/14484846.2015.1093231.

[5] C. Wang, Y. Fang, and S. Guo, "Multi-objective optimization of a parallel ankle rehabilitation robot using modified differential evolution algorithm," Chinese J. Mech. Eng. (English Ed., vol. 28, no. 4, pp. 702–715, 2015, doi: 10.3901/CJME.2015.0416.062.

[6] S. Rubaiee and M. B. Yildirim, "An energy-aware multiobjective ant colony algorithm to minimize total completion time and energy cost on a single-machine preemptive scheduling," Comput. Ind. Eng., vol. 127, no. January 2017, pp. 240–252, 2019, doi: 10.1016/j.cie.2018.12.020.

[7] K. Ishfaq et al., "Optimization of WEDM for precise machining of novel developed Al6061-7.5% SiC squeeze-casted composite," Int. J. Adv. Manuf. Technol., vol. 111, no. 7–8, pp. 2031–2049, 2020, doi: 10.1007/s00170-020-06218-5.

[8] T. A. Bhilawade, Advanced Engineering Optimization Through Intelligent Techniques, vol. 949. Springer Singapore, 2020.

[9] R. V. Rao, D. P. Rai, J. Ramkumar, and J. Balic, "A new multi-objective Jaya algorithm for optimization of modern machining processes," Adv. Prod. Eng. Manag., vol. 11, no. 4, pp. 271–286, 2016, doi: 10.14743/apem2016.4.226.

[10] K. K. Ingle and R. K. Jatoth, "An Efficient JAYA Algorithm with Lévy Flight for Non-linear Channel Equalization," Expert Syst. Appl., vol. 145, p. 112970, 2020, doi: 10.1016/j.eswa.2019.112970.

[11] R. V. Rao and R. B. Pawar, "Quasi-oppositional-based Rao algorithms for multi-objective design optimization of selected heat sinks," J. Comput. Des. Eng., vol. 7, no. 6, pp. 830–863, 2020, doi: 10.1093/jcde/qwaa060.

[12] R. H. Caldeira and A. Gnanavelbabu, "A Pareto based discrete Jaya algorithm for multi-objective flexible job shop scheduling problem," Expert Syst. Appl., vol. 170, no. January, p. 114567, 2021, doi: 10.1016/j.eswa.2021.114567.

[13] R. V. Rao, H. S. Keesari, P. Oclon, and J. Taler, "Improved multi-objective Jaya optimization algorithm for a solar dish Stirling engine," J. Renew. Sustain. Energy, vol. 11, no. 2, 2019, doi: 10.1063/1.5083142.

[14] R. V. Rao and A. Saroj, "Multi-objective design optimization of heat exchangers using elitist-Jaya algorithm," Energy Syst., vol. 9, no. 2, pp. 305–341, 2018, doi: 10.1007/s12667-016-0221-9.

[15] L. Lin and M. Gen, "Auto-tuning strategy for evolutionary algorithms : balancing between exploration and exploitation," pp. 157–168, 2009, doi: 10.1007/s00500-008-0303-2.

[16] S. Liu and M. Mernik, "Exploration and Exploitation in Evolutionary Algorithms : A Survey Exploration and Exploitation in Evolutionary Algorithms : A Survey ˇ," no. June, 2013, doi: 10.1145/2480741.2480752.

[17] E. Alba and B. Dorronsoro, "The Exploration / Exploitation Tradeoff in Dynamic," vol. 9, no. 2, pp. 126–142, 2005.

[18] K. Yu, J. J. Liang, B. Y. Qu, X. Chen, and H. Wang, "Parameters identification of photovoltaic models using an improved JAYA optimization algorithm," Energy Convers. Manag., vol. 150, no. August, pp. 742–753, 2017, doi: 10.1016/j.enconman.2017.08.063.

[19] R. Venkata Rao and H. S. Keesari, "Improved multi-objective Jaya optimization algorithm for a solar dish Stirling engine," Appl. Soft Comput. J., vol. 025903, no. November 2018, pp. 800–815, 2019, doi: 10.1063/1.5083142.

[20] S. Wang, F. Zhou, and F. Wang, "Effect of inertia weight ω on PSO-SA algorithm," Int. J. Online Eng., vol. 9, no. Specialissue.6, pp. 87–91, 2013, doi: 10.3991/ijoe.v9iS6.2923.

[21] Y. Gao, L. Peng, F. Li, M. Liu, and X. Hu, "Estimation of Distribution Algorithm with Multivariate T -Copulas for Multi-Objective Optimization," vol. 2013, no. February, pp. 63–69, 2013, doi: 10.4236/ica.2013.41009.

[22] K. Deb, A. Member, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm :," vol. 6, no. 2, pp. 182–197, 2002.

[23] N. K. Madavan, "Multiobjective optimization using a Pareto differential evolution approach," pp. 1145–1150.

[24] R. V. Rao, Jaya: An advanced optimization algorithm and its engineering applications. 2018.