International Journal of Engineering and Applied Physics (IJEAP)
Vol. 5, No. 1, January 2025, pp. 1136~1144
ISSN: 2737-8071

A Novel Hybrid Jaya Algorithm

Muhammad Islam[1], Muhammad Sulaiman[2], Amjad Khan[3], Atta Ur Rehman[4]
[1,2]Dept. of Mathematics at Abdul wali khan University Mardan, Pakistan.
[3]Dept. of Mathematics, Bacha Khan University Palosa, Charasadda, Pakistan.
[4]Dept. of Operation, Harbin Electric International Company (HEI), Shiekhpura, Pakistan.

Article Info

Article history:

Received Jun 9, 2024
Revised Nov 2, 2024
Accepted Nov 30, 2024

Keywords:

Evolutionary Algorithm
Differential Evolution
Fire Fly Algorithm
Shuffled Frog leaping
Gravitational Search Algorithm

ABSTRACT

The Hybrid JAYA (HJAYA) algorithm, a potent and hybridized optimization method, is suggested in this study as a solution to restricted design engineering optimization issues. The idea behind this innovative method is that the best solution found for a given problem shouldn't become stuck in local optima, but instead should aim to advance towards the best answers found thus far. The technique is further accelerated by using a novel starting strategy to provide better answers with fewer function evaluations. As fewer method-specific parameters are needed, this algorithm is simple to implement. By using it to resolve seven challenging constrained problems, including two from design engineering, the algorithm's effectiveness is assessed. Our findings are contrasted with those of other well-known methods found in the literature. The results show that, in terms of creating high-quality solutions, our suggested technique is either superior to or comparable to other algorithms. HJAYA is also applicable to issues in specific fields.

Corresponding Author:

Muhammad Islam,
Department of Mathematics,
Abdul Wali khan University,
Mardan, Pakistan.
Email: islammardanian603@gmail.com

1. INTRODUCTION

Evolutionary algorithms (EA) and swarm intelligence (SI)-based algorithms are two significant subgroups of population-based heuristic algorithms [1]. Examples of well-known evolutionary algorithms include Genetic Algorithm (GA), ES, EP, DE, BFO, AND AAI [2]. Well-known swarm intelligence-based algorithms include Particle Swarm Optimization PSO, SFL, ACO, ABC, FF and others. [3]. In addition to algorithms based on swarm intelligence and evolution, there are also more algorithms that use the fundamentals of many natural phenomena. Some notable examples within this category encompass the Gerande Explosion Method (GEM) [4]. Biogeography based Optimization (BBO) the Hormany Search (HS) technique, and the Gravitational Search Techniques (GSA).

The same controlling variables, such as population size, the number of generations, the size of the elite, etc., are needed for all probabilistic evolutionary and swarm intelligence algorithms. In addition to the standard control parameters, several algorithms require their own algorithm-specific control parameters. For instance, the PSO algorithm uses social, inertia weight, and cognitive parameters; the ABC algorithm uses the number of onlooker bees, scout bees, employed bees, and limit; and the HS algorithm uses harmony memory

consideration rate, pitch adjusting rate, and the number of improvisations. Similar to ES, EP, DE, BFO, AIA, SFL, ACO, and other algorithms.

The effectiveness of the aforementioned algorithms is greatly influenced by the proper tuning of the parameters that are individual to each method. When tuning algorithm-specific parameters incorrectly, the result is either an increase in processing effort. Rao et al. introduced the TLBO algorithm in light of this finding [5]. This does not call for any parameters relevant to the method. For the TLBO algorithm to function, just standard control parameters like population size and generational length are needed [6]. In 2015, R. Venkata Rao introduced the jaya algorithm, another algorithm-specific parameter-less method, in light of the TLBO algorithm's success [7].

The suggested technique has only one step and is significantly easier to use than the TLBO algorithm, which has two parts (the instructor phase and the learner phase). The Jaya algorithm operates very differently from the TLBO algorithm. The Jaya algorithm has been tested on a variety of constrained and unconstrained benchmark problems in addition to a number of case studies, including the optimization of surface grinding process parameters and the design of micro-channel heat sinks [8]. Several changes to the Jaya algorithm have also been suggested to further boost performance. Ocon and others [9] suggested that one of the top three best solutions be utilized to transition from the existing solution to a new one rather than employing the best option available to the present population. By including a self-adaptive weight strategy to alter the tendency of reaching the best solution and avoiding the worst solution, Yu et al. [10] suggested an Improved Jaya Optimization Algorithm. By incorporating the idea of adaptive populations, Rao and More [11] devised the Self-Adaptive Jaya Algorithm. By incorporating the idea of opposition-based learning into the fundamental Jaya algorithm [13], Rao and Rai [12] produced Quasi-Oppositional-Based an Algorithm.

2. OUR CONTRIBUTION

Due to the random numbers rand1 and rand2 in the search equation of jaya algorithm, sometime the algorithm stucks at local optima because when the pre-mature convergence occurs then the second part of search equation, which represents search around the best solution, become negligible due to this the random parameter rand1 does not work or multiplies random values to the second part in jaya equation which makes no sense when the population converges. So to get rid from this problem we have proposed a new hybrid jaya algorithm HJAYA in which we have replaced the random parameter rand1 by attractiveness factor β which is used in the firefly algorithm [14]. Additionally, five benchmark test problems and two sophisticated engineering design issues are used to evaluate the proposed approach. The outcomes demonstrated how much superior the suggested method is to other optimization strategies.

3. JAYA ALGORITHM

Remember that the objective function "f(x)" has to be maximized or minimized. Assume that at the end of the first iteration, there are "p" decision variables and "q" viable solutions. Assume that the worst candidate, "worst," decides the value of the function "f(u)" that is inferior to all other values of f(u), while the best candidate, "best," determines both. If at the ith iteration the candidates for k are Xj, k, and i, the value for variable j is updated [15].

According to Equation given:

$$X'_{j,k,i} = X_{j,k,i} + r1_{,j,i}\,(X_{j,best,i} - |X_{j,k,i}|) - r2_{,\,j,i}(X_{j,worst,i} - |X_{j,k,i}|) \quad (1)$$

It is approved when the updated new value X0 j, k, i of Xj, k, i delivers the optimum function value. All valid function values are kept at the end of an iteration and utilized as input values for the subsequent iteration. The algorithm JAYA's working mechanism is depicted in Fig. 1. The algorithm Jaya constantly struggles to move away from failure and approach success [16]. The algorithm works diligently to achieve the best solution in order to win, and for this particular reason it is called JAYA, which means triumph.

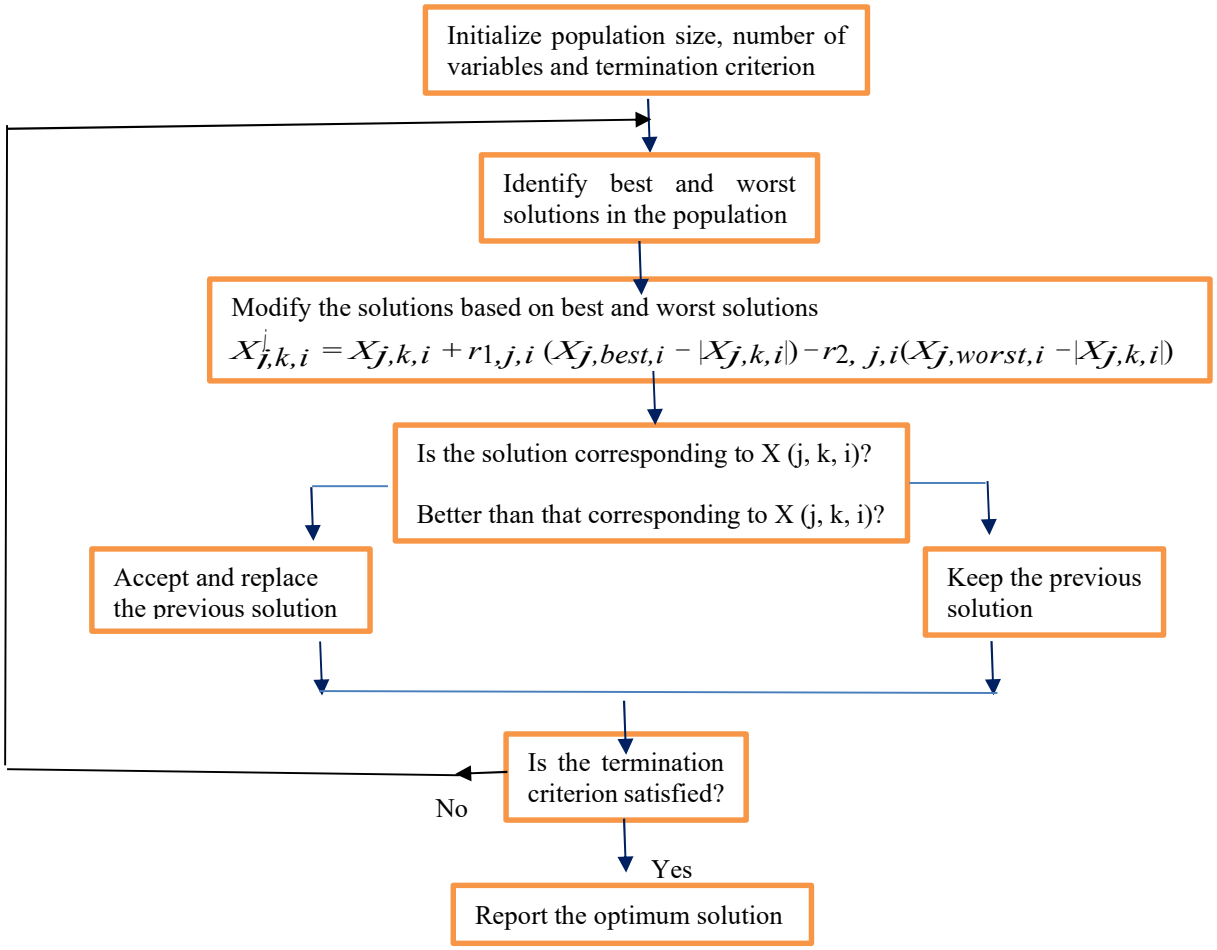

Figure 1. Flow chart of the JAYA Algorithm

3.1. FireFly Algorithm

FA was modelled by the behavior and flashing patterns of fireflies. Yang came up with FA. FA basically follows three idealized rules:

1) Since all fireflies are unisex, they are all attracted to one another.

2) A firefly's attractiveness is inversely correlated with how bright it appears to other fireflies, so for any two fireflies, the brighter one attracts the duller one, which causes the latter to migrate in its direction. If there are no brighter fireflies nearby, however, a firefly goes arbitrarily.

3) A firefly's brightness varies with the value of its objective function. The brightness and objective function in the maximization issue are inversely proportional. The formulation of appeal and variations in light intensity are the two fundamental components of FA. We can always assume, for convenience's sake, that the brightness of the fireflies or the intensity of the light, which is connected to objective function, determines how attractive they are. The brightness I of a firefly at a specific position x is typically taken as: f(x) I(x) for many optimization problems [17]. The distance rij between fireflies i and j affects how appealing they are to each other. Since light intensity diminishes with distance from the source and is also absorbed by the medium, appeal change with degree of absorption. The light intensity I(r) changes monotonically and exponentially with distance r.

It is expressed as:

$$I = Ioe^{-\gamma r} \qquad (2)$$

Here "Ɣ" is the coefficient of light absorption and I0 denotes, the light intensity originally as the attractiveness of the firefly is in proportional relation with light intensity judges by nearby fireflies, so here we are defining the attractiveness β of a firefly as:

$$\beta = \beta oe^{-\gamma r2} \qquad (3)$$

Where β0 represents the attractiveness at distance zero i.e r=0. It is important to note that one can replace the exponent ϒr by ϒr^m for m > 0. The systematic pseudo code of FA is given below:
The original Jaya Algorithm is:

Firefly Algorithm
Objective function f(x), x= (x1,..., xd)^T
Initialize a population of fire flies x_i (i = 1, 2 ……., n)
Define light absorption coefficient γ
While (t <MaxGeneration)
For i=1: n all n fireflies
For j=1: i all n fireflies
Light intensity I_i at x, is determined by f (x_i)
If (I_j > I_i)
Move firefly i towards j in all d dimensions
end if
Attractiveness varies with distance r via exp [-γr]
Evaluate new solutions and update light intensity
end for j
end for i
Rank the fireflies and find the current best end while
Postprocess results and visualization

$$Xj,k,i = Xj,k,i + r_1 , j, I (Xj, best, i - Xj,k,i - r2 j, i(Xj,worst, i - r2j.ij,worst,i - Xj,k,i) \quad (4)$$

Replacing r1 j, i by β we get:

$$Xj, k, i = Xj, k, i + \beta (Xj, best, i - Xj, k, i) - r2, j, i (Xj, worst, i - Xj, k, i) \quad (5)$$

In comparison to the original Jaya algorithm and other optimization methods, the newly acquired hybrid Jaya algorithm in Eq. (5) produces significantly better results. The convergence rate is accelerated and the optimization problem is better solved by the Brightness factor in Eq. 5.

4. RESULT

In this section, we'll talk about the outcomes of two engineering design issues and five benchmark test problems using the HJAYA Algorithms.

4.1 Problem: 01 Welded Beam Design Problem

This issue's objective is to reduce the price of welded beams. It is possible to formulate the welded beam design problem's objective function F(X), which is totally cost-based and includes welding labour, set-up, and material costs, as follows in Fig 2 [18].

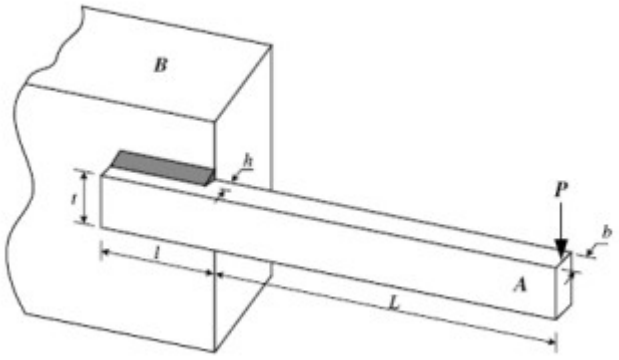

Figure 2. Welded beam design

The welded beam design problem is addressed with the newly presented algorithm, and the best result is attained after just 20 separate runs. In Table 1, you can see the solution vector, constraints, and value of the objective function [19].

Table 1. Optimal outcomes achieved through the HJAYA Algorithm for welded beam problem

parameter	x_1	x_2	x_3	x_4
values	0.206244456922346	3.45941331041609	9.03700769940361	0.205731448108665
parameter	g_1	g_2	g_3	g_4
values	-0.3692	-2.8117	0.0000	-3.4339
parameter	g_5	g_6	g_7	f
value	-0.2355	-0.3258	-0.0812	1.724234455494711

Table 2. Evaluating the outcomes of each algorithm in the context of the welded beam problem

Method	x_1	x_2	x_3	x_4	Best
GA	0.2088	3.4205	8.9975	0.2100	1.7483
PSO	N/A	N/A	N/A	N/A	1.922
CPSO	0.202369	3.544214	9.04821	0.205723	1.728024
COPSO	0.205730	3.470489	9.036624	0.205730	1.724852
CDE	0.203137	3.542998	9.033498	0.206179	1.733461
FA	0.2015	3.562	9.0414	0.2057	1.731207
BB-BC	0.205718	3.4709118	9.0364603	0.205737	1.724928
DBB-BC	0.205730	3.4704887	9.0366239	0.205730	1.724852
Proposed hybrid Jaya	0.20624	3.45941	9.03701	0.20573	1.724235

It is evident from Table 2 that the HJAYA algorithm's result is superior to the solutions produced by all other algorithms.

4.2 Problem: 02 Spring Design Problem

As seen in Fig. 3, [20], the goal of this challenge is to reduce the weight of a spring. Minimum deflection, Shear stress, and surge frequency are a few of the restrictions that apply to the minimizing procedure. The three variables in this problem are wire diameter (d), mean coil diameter (D), and the number of active coils (N). The spring design challenge is formulated as follows:

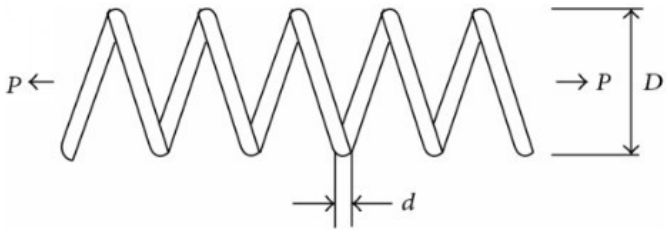

Figure 3. Illustrative chart of the spring design issue

The following is a mathematical expression of the spring design problem:

$$_{M3in}\, f(X) = x_2 x^2 (x_1 + 2) \qquad (6)$$

Bounds on the variables are:

$$0.050 \le x_1 \le 2.00,\ 0.250 \le x_2 \le 1.300,\ 2.00 \le x_3 \le 15.00.$$

Table 3. Optimal outcomes achieved through the HJAYA Algorithm forspring design problem

Parameter	x_1	x_2	x_3	f
Values	0.051838	0.360328	11.080204	0.012665085
Parameter	g_1	g_2	g_3	g_4
Values	-0.0000	0.0000	-4.0609	-0.7252

Table 4. Contrast the solution achieved in the spring design problem byproposed HJAYA Algorithm with other algorithms

Algorithm	x_1	x_2	x_3	**Optimum value**
HYBRID JAYA	**0.051838**	**0.360328**	**11.080204**	**0.012665085**
GWO	0.051690	0.323680	13.525410	0.0127022
GSA	0.050276	0.323680	13.525410	0.0127022
PSO	0.051728	0.357644	11.244543	0.0126747
ES	0.051989	0.363965	10.890522	0.0126810
GA	0.051480	0.351661	11.632201	0.0127048
HS	0.051154	0.349871	12.076432	0.0126706
DE	0.051609	0.354714	11.410831	0.0126702
Mathematical optimization	0.053396	0.399180	9.1854000	0.0127303
Constraint correction	0.050000	0.315900	14.250000	0.0128334

4.3 Constraint Benchmark Test Problem

We can observe from the preceding sections that the suggested algorithm outperforms other algorithms in solving engineering design challenges [21, 22]. The proposed method will now be used in this section to solve benchmark test problems, and the results will then be compared to those of other algorithms.

4.3.1 Problem: 03 Test Problem G04

Mathematically G04 can be written as:

$$\min f(u) = 5.35785470u^2 + 0.83568910u_1 u_5 + 37.293239 u_1 - 40792.1410,$$

Subjected to:

$H_1(b) = v(b)-92 \leq 0,\ h_2(b) = v(b) \leq 0,\ h_3(b) = u(b) - 110 \leq 0,\ h_4(b) = u(b) +90 \leq 0,\ h_5(b) = x(b) -25 \leq 0,$

$H_6(b) = -x(b)+20 \leq 0.$

For this issue G04, the proposed HJAYA Algorithm yielded the optimal value and solution vector, which are shown in Table 5.

Table 5. Best solution obtained by proposed HJAYA Algorithm for G04

Solution vector	u_1	u_2	u_3	u_4	u_5
Values	78.0027	33.0000	29.9948	45.0000	36.7743
Objective function	$f(u)$				
Value	-30665.613				

4.3.2 Problem: 04 Test Problem G06

The mathematical form of benchmark test problem G06 is given below.

$$\min f(b) = (b1-10)^3 + (b2-20)^3$$

Subjected to

$$h_1(b) = (b1-5)^2 + (b2-5)^2 + 100, \; h_2(b) = (b1-5)^2 + (b2-5)^2 - 82.81 \leq 0.$$

Where

$$1b = (13, 0) \text{ and } ub = (100, 100)$$

For this issue G06, the proposed HJAYA Algorithm yielded the optimal value and solution vector, which are shown in Table 6.

Table 6. Optimal outcomes achieved through the HJAYA Algorithm for problem G06.

Solution vector	b_1	b_2
Values	14.094964107721200	0.842880980693446
Objective function values	$f(b)$ -6961.904	

4.3.3 Problem: 05 Test Problem G08

Problem G08 from the benchmark test has the following mathematical form:

$$\text{Max } f(a) = \sin^3(2\pi a1)\sin(2\pi a2) \,/\, a^3(a_1 + a_2)$$

For this issue G08, the proposed HJAYA Algorithm yielded the optimal value and solution vector, which are shown in Table 7.

Table 7. Best solution obtained by proposed HJAYA Algorithm for G08

Solution vector	a_1	a_2
values	1.22797135333232	4.24537336684669
Objective function	$f(a)$	
Value	-0.0958	

4.3.4 Problem: 06 Test Problem G09

The mathematical form of G09 test problem is given by

$$\text{Minf}(b) = (b1 - 10)^2 + 5(b2 - 12)^2 + b^43 + 3(b4 - 11)^2 + 10b^6_5 + 7b^2_6 + b^4_7 - 4b_6\, b_7 - 10b_6 - 8b_7$$

Subjected to:

$$h1(b) = 2b^2_1 + 3b^4_2 + b_3 + 4b^2_4 + 5b_3 - 127 \le 0, \quad h2(b) = 7b_1 + 3b_2 + 10b^2_3 + b_4 - b_5 - 282 \le 0,$$

$$h3(b) = 23b_1 + b^2_2 + 6b^2_5 - 8b_7 - 196 \le 0, \qquad ha(b) = 4b^2_1 + b^2_2 - 3b_1\, a_2 + 2b^2_3 + 5b_6 - 11b_7 \le 0,$$

For this issue G09, the proposed HJAYA Algorithm yielded the optimal value and solution vector, which are shown in Table 8.

Table 8. Optimal outcomes achieved through the HJAYA Algorithm for problem G09.

Solution vector	b_1	b_2	b_3	b_4	b_5	b_6	b_7
Values	2.2930	31.9835	-0.7515	4.2970	-0.6141	0.7128	1.4562
Objective function value	$f(b)$ 681. 234						

4.3.5 Problem: 07 Test Problem G11

The mathematical expression representing the G11 test problem is provided below:

$$\min f(b) = bl^2 + (b2 - 1)^2 \qquad b_2 - bl^2 = 0.$$

For this issue G11, the proposed HJAYA Algorithm yielded the optimal value and solution vector, which are shown in Table 9.

Table 9. The optimal outcomes achieved through the HJAYA Algorithm for problem G11.

Solution vector	b_1	b_2
Values	-0.706709508428322	0.499452546261091
Objective function	$f(b)$	
Value	0.749986082747506	

The results of the suggested method for test problems G11, G09, G08, G06, and G04, are compared to those of other algorithms in the accompanying Table 10. The values highlighted in bold are the results of using the hybrid Jaya algorithm to solve several test problems.

Table 10. Comparison of HJAYA with other algorithms

Algorithms/problems	G04	G06	G08	G09	G11
Hybrid JAYA	-30665.613	-6961.904	-0.0958	680.1861	0.749986
HM	-30664.5	-6952.1	-0.095825	680.91	0.75
GA	-30626.053	-6952.472	-0.095825	685.994	0.75
BBO	-30665.539	-6961.8139	-0.095825	680.6301	0.7499
ABC	-30665.539	-6961.814	-0.095825	680.634	0.75
DE	-30665.539	-6954.434	-0.095825	680.63	0.752
Jaya	-30665.5387	-6961.814	-0.095825	680.6301	0.7499
PSO	-30665.539	-6961.814	-0.095825	680.63	0.749

5. CONCLUSION

Most likely, in addition to modifying the algorithm's general regulating parameters, the complete swarm intelligence and evolutionary algorithm employed some tuning for unique algorithm parameters. The optimisation of a certain algorithmic parameter affects the algorithm's performance. A novel algorithm called TLBO was introduced last year; it just employs the common regulating parameter of all algorithms and does

not require any additional particular parameters. In light of this success in TLBO, let's discuss optimisation literature. Recently, the well-known Java algorithm—which has less algorithm-specific parameters—was proposed. The Java algorithm is quite straightforward and simple to use. For engineering design challenges and other test problems, the Java approach produces results that are significantly superior to those produced by other optimisation methods. In this research, we have created a hybrid version of the java algorithm by substituting the attractiveness factor for the java algorithm's random number generator, rand1. The welded beam design and spring design issues, as well as five benchmark test problems, are applied to the suggested hybrid java algorithm.

The results of different optimisation algorithms, are compared to the results of our hybrid Java algorithm. For these issues, use mathematical optimisation, constraint correction, HM, BBO, and ABC. The introduction to optimisation, various forms of optimisation, and optimisation algorithms were actually covered in this article. A Java algorithm for confined and unconstrained optimisation problems has also been discussed. The outcomes of the proposed Java algorithm and other optimisation strategies are then compared.

REFERENCES

[1] Eiben, A.E. and Rudolph, G., 1999. Theory of evolutionary algorithms: a bird's eye view. Theoretical Computer Science, 229(1-2), pp.3-9.

[2] Li, M.J., Luo, A. and Tong, T.S., 2004. Artificial immune algorithm and its applications. Kongzhi Lilun yu Yingyong/Control Theory and Applications (China), 21(2), pp.153-157.

[3] Geem, Z.W., Kim, J.H. and Loganathan, G.V., 2001. A new heuristic optimization algorithm: harmony search simulation, 76(2), pp.60-68.

[4] Ahrari, A. and Atai, A.A., 2010. Grenade explosion method—a novel tool for optimization of multimodal functions. Applied Soft Computing, 10(4), pp.1132-1140.

[5] Sarzaeim, P., Bozorg-Haddad, O. and Chu, X., 2018. Teaching-learning-based optimization (TLBO) algorithm. In Advanced Optimization by Nature-Inspired Algorithms (pp. 51-58). Springer, Singapore.

[6] Rao, R.V. and More, K.C., 2015. Optimal design of the heat pipe using TLBO (teaching–learning-based optimization), algorithm. Energy, 80, pp.535-544

[7] Rao, R., 2016. Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. International Journal of Industrial Engineering Computations, 7(1), pp.19-34.

[8] R V. Rao, D.P. Rai, J. Balic, Surface Grinding Process Optimization Using Jaya Algorithm, Comput Intell Data Min. 2 (2016) 487-495. doi:doi.org/10.1007/978-81322- 2731-1-46

[9] Ocłoń, P. Cisek, M. Rerak, D. Taler, R.V. Rao, A. Vallati, M. Pilarczyk, Thermal performance optimization of the underground power cable system by using a modified Jaya algorithm, Int J Therm Sci. 123 (2018) 162–180.

[10] K. Yu, J.J. Liang, B.Y. Qu, X. Chen, H. Wang, Parameters identification of photovoltaic models using an improved JAYA optimization algorithm, Energy Convers Manag. 150 (2017) 742–753. Doi: 10.1016/j.enconman.2017.08.063.

[11] R. V. Rao, K.C. More, optimal design and analysis of mechanical draft cooling tower using improved Jaya algorithm, Int J Refrig. 82 (2017) 312–324. Doi: 10.1016/j.ijrefrig.2017.06.024.

[12] R.V. Rao, D.P. Rai, Optimization of welding processes using quasi-oppositional based Jaya algorithm, J Exp Theor Artif Intell. 29 (2017) 1099–1117. doi:10.1080/0952813X.2017.1309692.

[13] R.V. Rao, A. Saroj, constrained economic optimization of shell-and-tube heat exchangers using Elitist-Jaya algorithm, Energy. 128 (2017) 785–800. Doi: 10.1016/j.energy.2017.04.059.

[14] Yang, X.S., 2010. Firefly algorithm, stochastic test functions and design optimization. ArXiv preprint arXiv: 1003.1409.

[15] Deb, K., 1991. Optimal design of a welded beam via genetic algorithms. AIAA journal, 29(11), pp.2013-2015.

[16] W. S. Sakr, R. A. El-Sehiemy, and A. M. Azmy, "Adaptive Differential Evolution Algorithm for Efficient Reactive Power Management", Applied Soft Computing, vol. 53 pp. 336-351, 2017.

[17] H. Singh and L. Srivastava, "Modified differential evolution algorithm for multi-objective VAR management." International Journal of Electrical Power & Energy Systems vol. 55, pp. 731-740, 2014.

[18] W. Zhang and Y. Z. Liu, "Multi-objective reactive power and voltage control based on fuzzy optimization strategy and fuzzy adaptive particle swarm." International Journal of Electrical Power & Energy Systems vol. 30, no.9, pp. 525-532, 2008

[19] W. Yan, S. Lu and D. C. Yu, "A novel optimal reactive power dispatch method based on an improved hybrid evolutionary programming technique." IEEE Transactions on Power Systems vol. 19, no.2, pp. 913-918, 2004.

[20] R. H. Liang, J. C. Wang, Y. T. Chen and W. T. Tseng, "An enhanced firefly algorithm to multi-objective optimal active/reactive power dispatch with uncertainties consideration." International Journal of Electrical Power & Energy Systems, vol. 64, pp. 1088-1097, 2015

[21] D. Devaraj and J. P. Roselyn. "Genetic algorithm based reactive power dispatch for voltage stability improvement." International Journal of Electrical Power & Energy Systems vol. 32, no. 10, pp. 1151-1156, 2010.

[22] W. S. Sakr et al. "Efficient reactive power management via enhanced differential evolution algorithm with adaptive penalty factor." Int. J. Power Eng. Energy vol. 6, no. 2, pp. 542-550, 2015.