IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Mathematical Modelling of Engineering Problems
Vol. 9, No. 2, April, 2022, pp. 507-514
Journal homepage: http://iieta.org/journals/mmep

Solving Symmetrical Drop Suspended Equilibrium Equation by Artificial Bee Colony Programming

Amel Serrat*, Bachir Djebbar

Department of Computer Science, Faculty of Mathematics and Computer Science, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, USTO MB, BP. 1505 El M'naouer, Oran 31000, Algeria

Corresponding Author Email: amel.serrat@univ-usto.dz

https://doi.org/10.18280/mmep.090229

Received: 21 October 2021
Accepted: 20 February 2022

Keywords:
artificial bee colony programming, differential equation, symbolic regression, suspended symmetric drop

ABSTRACT

In this paper we study the aptitude of Artificial Bee Colony Programming (ABCP) to solve a Suspended Symmetric Drop Equilibrium Equation (SSDEE) described by a non linear second order differential equation. Artificial Bee Colony Programming was developed by Boudouaoui and Karaboga in 2020 to extend Artificial Bee Colony for symbolic regression problems as Genetic Programming (GP) where food source represents tree.The Suspended Symmetric Drop Equilibrium Equation is represented by a second order non linear Differential Equation with boundary conditions. The solving process is fully symbolic, and The Artificial Bee Colony Programming aims to find the best solutions according to fitness. We apply it with different parameters adjustment and we also apply the 4th Runge-Kutta method as usual method. The Artificial Bee Colony Programming solution was compared with the Fourth Runge-Kutta(4RK) and we found that Artificial Bee Colony Programming can solve Suspended Symmetric Drop Equilibrium Equation with a bite agreement and with fewer constraints.

1. INTRODUCTION

Most real-world physical phenomena are modelled by differential equations and require solutions. Differential equations can be simple, complex or computationally time consuming [1-7]. The complexity of solution varies from one equation to another depending on whether it is ordinary or partial, linear or non-linear, with or without initial values or boundary conditions and depending on the order of the equation [3-7]. As a result, in this work, we present an artificial intelligence-based optimization and approximation technique for solving an ordinary nonlinear second order differential equation with boundary conditions.

The approach is known as Artificial Bee Colony Programming (ABCP), and it is based on Darwinian evolutionary principles such as 'natural selection and competition' [8-11]. The Artificial Bee Colony technique mimics the collective behavior of foraging bees [9, 11, 12] as they seek honey and pollen.

Many studies have increased the effectiveness and accuracy of the artificial bee colony method. Many changes in representation and design have been made to the Artificial Bee Colony Method, allowing it to function better. The performance of the artificial bee colony method has been demonstrated in a variety of applications involving high-complexity combinatorial problems in a variety of domains [13]. We can mention here image &video processing [14-18], data clustering [19-21], path planning [22], localization [23, 24], Scheduling [25-30], data-mining [31, 32] and symbolic regression [11, 33] and these are some of the fields.

The Artificial Bee Colony Programming technique, created by Karaboga and Ozturk in 2012 [9, 11, 12] to extend Artificial Bee Colonies, exhibits the same symbolic regression as Koza's Genetic Programming [34, 35]. The depiction of each individual in the latest version of The Artificial Bee Colony is represented by a tree [11]. Each individual in the foraging bee population corresponds to one of the problem's possible solutions, and the algorithm remains the same as that of Karaboga's artificial bee colonies developed in 2005 [11]. In 2020, Boudouaoui et al. [36] used Artificial Bee Colony Programming on some examples of differential equations and integrated the boundary conditions to the fitness function calculation [36-38].The constraint integration was constructed using the ordinary differential equation $f(x, y, y^1, \ldots, y^n) = 0$ where $f$is a given function, .and $y$is an unknown function of x and the boundary conditions $g(x, y, y^1, \ldots, y^{n-1}) = 0$ where $g$is a function that covers conditions:

$$fitness_i = \sum_j f\left(x_j, y_j\left(x_j\right), y_j^1\left(x_j\right), \ldots, y_j^i\left(x_j\right)\right)^2 + \lambda \sum_k abs\left(g_k\left(x, y_i, y_i^1, \ldots, y_i^{n-1}\right)\right) \quad (1)$$

With λ positive.

In this article, we propose using Artificial Bee Colony Programming to get a symbolic solution in some closed form to a Differential equation. The Artificial Bee Colony Programming try solve the equilibrium profile equation of a suspended drop on a solid wall, which is expressed as a nonlinear ordinary differential equation [37, 38]. Extensive trials are carried out, and the findings demonstrate that the approach is capable of achieving good performance.

The article is divided as follows: section 2 contains a description of the fundamental Artificial Bee Colony algorithm, and section 3 encloses a detailed discussion of the

Artificial Bee Colony Programming method. Section 4 discusses the use of Artificial Bee Colony Programming and the findings achieved.

2. RELATED WORK

In this section, we introduce the associated work of our study, first presenting some related work regarding Differential Equation resolution.

We begin with analytical procedures [39] such as variable separation and integration, which yield reliable answers; and they try to find a solution in a closed form, where a finite number of elementary functions are used to express the solution. Nevertheless, these methods cannot solve all differential equations, and rewriting equations in a closed form takes time and requires a large mathematical background.

Numerical techniques [40] are an alternative to analytical methods which are simple to apply, yield an approximate solution, and are:

- based on the notions of regression and interpolation.
- the answer may not be approximated, or not analytical (which means a continuous and differentiable) and the quality of the solution remains questionable.
- specific to some differential equations.

We can mention some of the numerical methods [40] like: finite difference, Runge Kutta, series expansion, divided difference and finite element and the Table 1 represents the most of them.

Optimization methods are another alternative to analytical methods like machine learning and evolutionary methods. We can mention:

Neural Net: it was the first technique [41], developed by Lee in 1989 as a model for the differential equation, is artificial neural networks. The applications are then based on error backpropagation. Shi and Xu [41] used it on several differential equations.

Deeplearning Neural Net: based on neural Net with complex training process and different learning method [42, 43]. The method was applied to high dimensional partial Differential equation [42], and in Ref. [43] to a system of differential equation with irregular solution.

Fuzzy Logic: based on fuzzy inference scheme [44]. It was applied to partial differential equation [44].

LS-SVM: based support Vector machine and least square method [45] was applied to high order ordinary differential equation.

Table 1. Numerical methods set

Finite Difference Method	Runge Kutta Method	Prediction & correction method
Series Expansion Method	Order Reduction Method	Extrapolation method
Finite Element	Galerkin/ Bubnov Method	Collocation Method
Variable Step Method	Shwarz Method	Variable order Method

Differential Evolution: based on population of candidate that optimizes a given problem [46, 47]. It was applied to Lorenz and Vander Pol differential equation, and to Partial Differential Equation.

Genetic Algorithm: based on evolutionary mechanism [48, 49]. It was applied to ordinary Differential Equation [48] and Delay differential equation [49].

Genetic Programming: based on Genetic Algorithm and Grammatical Evolution [50] in order to get a symbolic solution. It was applied [50] to several Differential Equation.

Ant Colony Programming: based on Artificial Ant Colony and Grammatical Evolution. It was applied Second Order Differential Equation [51].

3. ARTIFICIAL BEE COLONY METHOD

Artificial Bee Colony is a method of optimization based on bee swarms. It was created by Karaboga [11, 12] to imitate the collective social behaviour of foraging bees while searching for pollen and nectar.

This behavior is governed by a single member of the bee colony, the worker. This worker performs several tasks during her life as in Figure 1.

We are only interested in two activities in this article: foraging (the exploration phase) and pollen and nectar collecting (the exploitation process), both of which are impacted by the following elements [37]:

(1) Weather conditions
(2) The food source's location respect to the hive
(3) Reliability (sugar and protein concentration)
(4) The amount of nectar and pollen

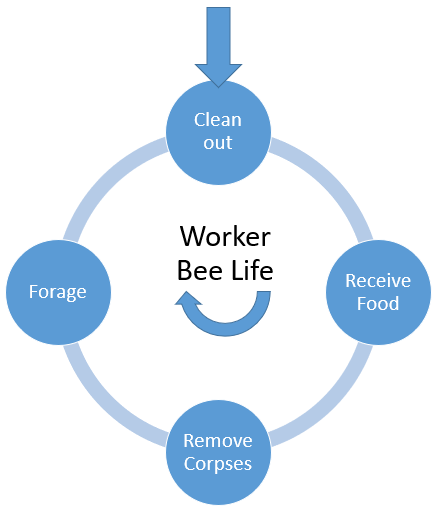

Figure 1. Worker bee tasks

3.1 The basic behavior

The artificial bee colonies are made up of three sorts of worker bees: Scout, Employed, and Onlooker, and each type corresponds to a different processing phase in the algorithm.

The Scout bee searches for food sources X (potential solutions), and when discovered, it becomes an Employed bee.

The employed bee will compute the concentration of sugar and protein in a food source, the amount of nectar and pollen, and the location relative to the sun. In addition to the calculations, a sample of nectar and pollen is taken.

She returns to the hive after gathering this knowledge and shares it with her comrades, the Onlooker bees.

Employed bee performs a dance and allows the Onlooker bees to sniff the meal sample. The dance might be rapid or slow to notify them of the quantity, and the posture of the employed bee will indicate the location of the meal as in Figure 2 [11-33].

The onlooker bee watches the dance and detects the fragrance of the food before flying off to find and collect nectar and pollen [8-33].

The algorithm for artificial bee colonies is divided into three stages [11-33] as in Figure 3.

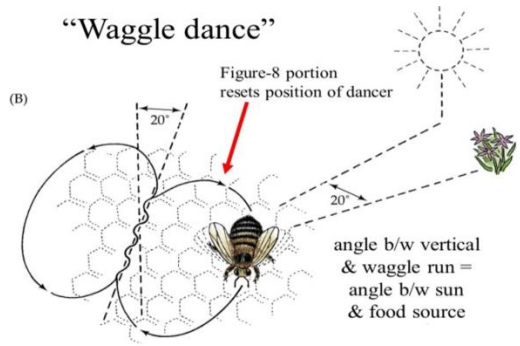

Figure 2. The Employed bee dance

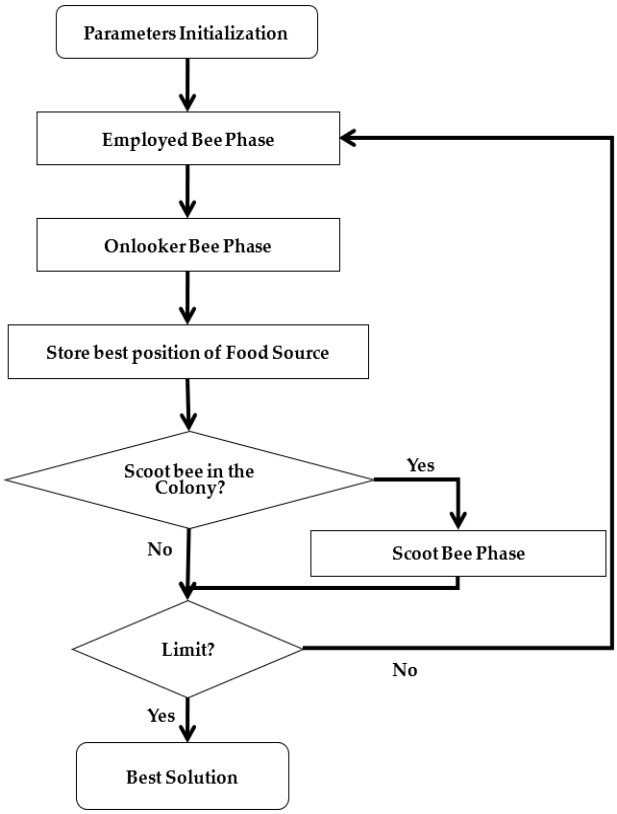

Figure 3. The Ant Bee Colony flowchart

3.2 The basic operations in ABC method

The basic operations in Artificial Bee Colony Programming (ABCP) are: parameters initialization, tree construction, Fitness evaluation, information sharing, differentiation, and greedy selection [34-38]. We can report each operation and its requirements as follows:

Table 2. Terminals set

Terminals	
Constant	0,1,2,3,4,5,6,7,8,9
Variables	t

Table 3. Functions set

Functions	
Arithmetic functions	Sin, cos, exp, log, sqrt
Operators	+,-,*,/,(,)

3.2.1 Parameters initialization

The parameters are as follows [34-38]:

The Terminals collection. The terminal set corresponds to the content of the tree's leaves. The set of terminals is mention in Table 2.

It is made up of the variables and constants of the differential equation and is determined by the problem at hand as mention in Table 2.

The Functions collection. Except for the leaves, functions match to the content of the tree's nodes [34-38].

It is made up of mathematical functions such as addition, subtraction, division, multiplication, and other more complicated functions as in Table 3, and the set also varies depending on the issue being solved [34-38].

The Fitness function. The function assigns a value to each tree in the population. This procedure enables the algorithm to proceed with the individual selection (the strongest element of the population) [34-38].

The free parameters of control are:

(1) The population size of solutions (PS)
(2) The maximum number of cycles (Cycle)
(3) The initial maximum depth of the tree
(4) The number of Food Sources (FN)
(5) The two coefficients β and γ indispensable for the diversification of the individuals
(6) The limit parameter is a counter for each individual described to the lifetime of a source (individuals).

3.2.2 Tree construction

Random tree construction [11] in ABC is a crucial operation in this method and can affect its performance. This operation is done by one of the following methods:

The Grow method. The leaves on trees [11, 52-55] created by the Grow technique are not all at the same depth (see Figure 4).

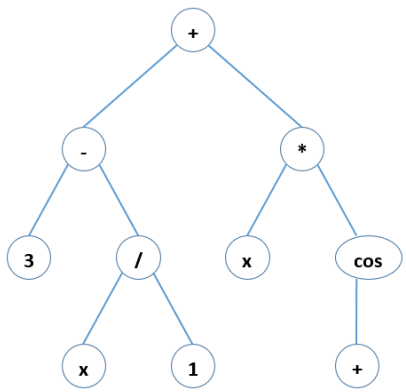

Figure 4. Tree created by the Grow method

The Full method. The process is similar to grow method with an exception that if the depth is less than the maximum allowable tree depth, a node is picked at random from a set of internal nodes rather than from the set of terminals. The leaves of the trees [11, 34-38, 52-53] created using the complete technique are all positioned at the same depth (see Figure 5).

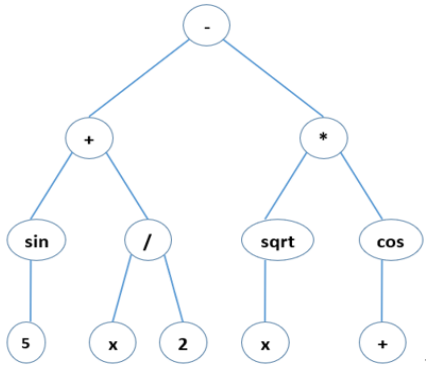

Figure 5. Tree created by the Full method

The Ramped Half&Half. If the maximum depth of the tree [52-55] is **N**, the Ramped method will only allow for the creation of trees ranging in size from **02 to N**, whereas the half and half method will ensure equity between the Grow method and the FULL method by generating **50%** of the tree using the Grow method and the other half using the FULL method.

3.2.3 Fitness evaluation

The fitness value for each solution [11, 36-38] is calculated as in Eq. (2):

$$fit(X_i) = \begin{cases} \dfrac{1}{1+f(X_i)}, si\ f(X_i) \geq 0 \\ 1 + abs\big(f(X_i)\big), else \end{cases} \quad (2)$$

where the evaluation of a solution is calculated as follows:

$$f(X_i) = \frac{1}{2}\sum_{j=1}^{N}(Desired_i - obtained_i)^2 \quad (3)$$

3.2.4 Sharing information

The Onlooker and Employed bees [11, 36-38] exchange information about the quantity and quality of nectar and pollen (X_i) in the symbolic manner (see Figure 6):

(1) Select randomly a solutionX_ka neighbor of X_i;
(2) Select randomly a solution X_k a neighbor of X_i;
(3) Choose two nodes at random, the first on X_i and the second on X_k, with a Coefficient β to have a function and γ to have a terminal;
(4) $V_k$is obtained by replacing the first subtree on X_i with the first subtree on X_k.

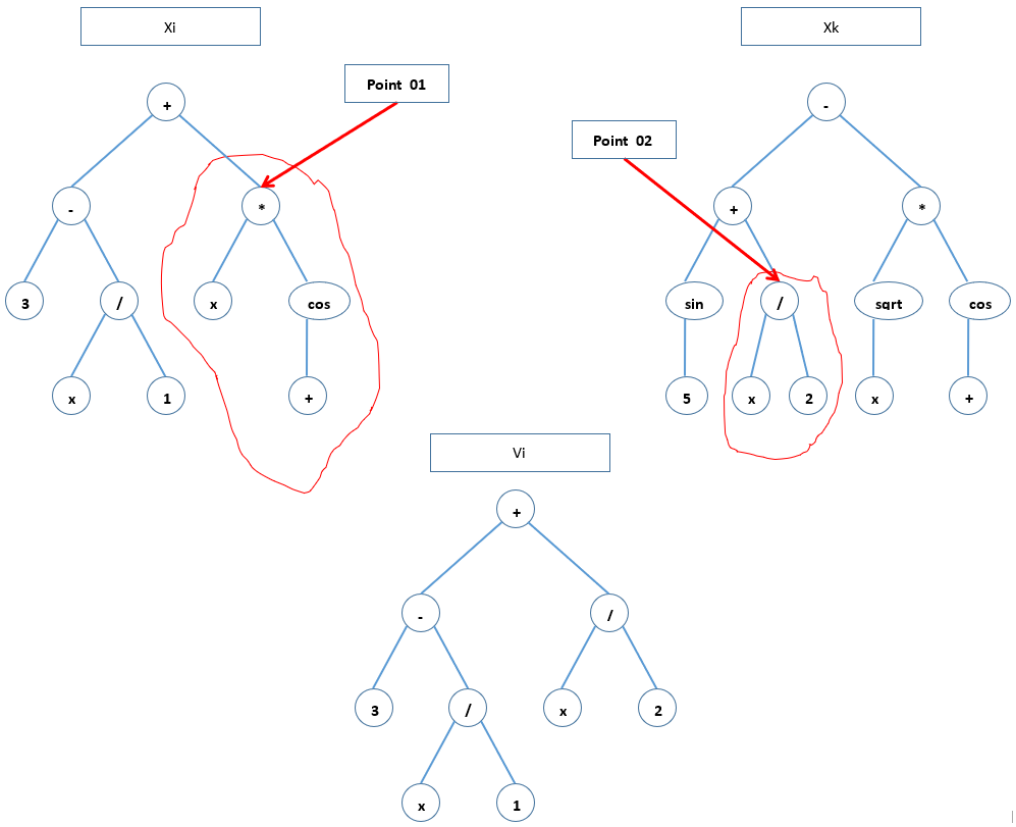

Figure 6. The Sharing information process

3.2.5 Differentiation

The ABC method guarantees [36-38] population variety by raising or lowering the odds during roulette selection by a probability.

This probability as in Eq. (4) is allocated [36-38] to each member of the population and is modified with each generation.

$$P(X_i) = \beta * \left(\frac{fit(X_i)}{max(fit)}\right) + \gamma \quad (4)$$

3.2.6 Greedy selection

The aim of the Greedy selection [36-38] is to keep only the best and the choice is made as follows:

$$if fit(V_k) > fit(X_i) then \begin{cases} replace\ X_i\ by\ V_k \\ compt_i = 0 \end{cases} \quad (5)$$
$$else\ compt_i = compt_i + 1$$

3.3 The ABCP process

The Artificial Bee Colony Programming (ABCP) process uses three-phase: Bee-Employed, Bee-Onlooker, and Bee-Scout phases [11, 36-38]. These phases represent the various worker bee tasks involved in the search for a food source. We can report each phase and its requirements as follows:

3.3.1 Bee employed phase

Create a nearby solution [11] V_k for each solution or tree X_i of the population X using the previously described information sharing method, compute its fitness, and then choose between X_i and V_k (see Figure 7).

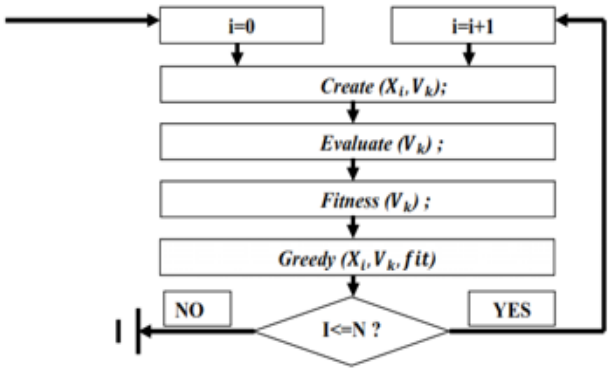

Figure 7. The bee employed phase

3.3.2 Bee onlooker phase

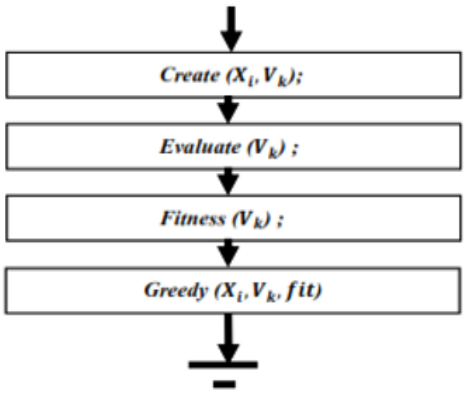

Figure 8. The bee onlooker phase

The Bee Onlooker phase begins after a probability calculation for each individual in the population [36-38].

At random number generator [8] selects an individual X_i from the population X, the information sharing mechanism creates an adjacent solution V_k, the fitness for this new individual V_k is computed, and a choice between X_i and V_k is made (see Figure 8).

The Bee-Onlooker phase [11] can be seen as a single iteration of the Employed process where the individual X_i is chosen by the roulette selection.

3.3.3 Bee scout phase

If a solution X_i is not visited or updated for a certain period of time, it is considered abandoned and must be destroyed, [11] with a new one generated at random using the Grow technique.

The **limit** parameter formalizes the duration. Alternatively, if the variable **compt** approaches the **limit**, it will be reset to zero (see Figure 9).

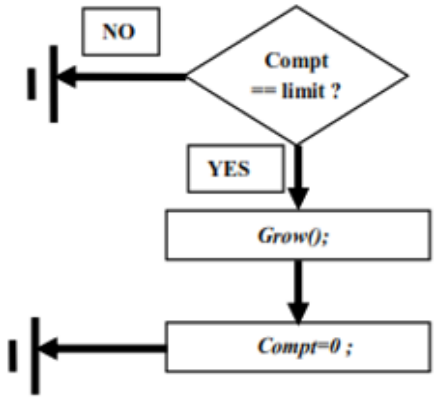

Figure 9. The bee scout phase

4. EXPERIMENTAL RESULTS & DISCUSSION

The Artificial Bee Colony method is implemented to solve a second order non linear differential Equation.

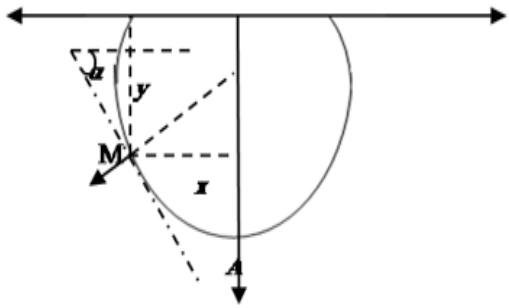

Figure 10. A Suspended Drop on a solid wall

The equilibrium equation of the drop as in Eq. (6) describes the profile of a drop of Decane suspended as in Figure 10 from a Pyrex wall and dipped in salt water [51]:

$$C_1 - C_2 + \frac{f^2(y)}{(1+f'(y)^2)^{3/2}} = \frac{1}{f(y)(1+f'(y)^2)^{1/2}} \quad (6)$$

Under boundary conditions:

$$\begin{cases} f(0) = x \\ f'(0) = -cotg(g\alpha) \end{cases} \quad (7)$$

where,

$$C_1 = \frac{2}{b} + \frac{\Delta\rho g}{\sigma} h \text{ and } C_2 = -\frac{\Delta\rho g}{\sigma} \quad (8)$$

And the other drop parameters are mentioned in Table 4. The results of the Artificial Ant Colony programming method are compared with the Runge-Kutta method.

The Artificial Bee Colonies method is tested with different values of the following parameters: population size, the two degrees of visibility importance β, γas shown in Table 5.

Table 4. Drop profile parameters

Pyrex/Décane	
ρ_1	0.7268
ρ_2	1.0196
η	8.5
V	47.33
σ	60.53
θ_1, θ_2	56;56

Table 5. Comparison results of ABCP under different variatio

Parameters	N°	Value	MSE	Variance
FN=PN	01	30	2,400776E-03	1,107680E-03
	02	40	2,234350E-03	2,353120E-03
	03	50	2,726163E-02	2,044090E-03
	04	60	1,774828E-03	2,073070E-04
	05	70	2,424597E-03	1,479500E-04
	06	80	2,004544E-02	1,828750E-03
	07	100	2,375986E-03	2,570440E-04
	08	150	**1,611740E-04**	**2,054620E-05**
	09	200	1,816470E-03	2,703080E-04
	10	300	1,772671E-03	2,410520E-04
	11	400	2,533931E-03	3,496970E-04
β	12	0.1	2,079761E-03	3,270530E-03
	13	0.2	1,948729E-03	3,844950E-03
	14	0.4	1,772671E-03	2,410520E-04
	15	0.5	2,055930E-03	4,189800E-04
	16	0.6	**1,705610E-04**	**5,139720E-05**
	17	0.7	2,178042E-03	3,909670E-04
	18	0.8	2,293738E-03	3,945580E-03
	19	0.9	2,473200E-03	4,878640E-03
	20	01	2,102692E-03	3,281240E-03
γ	21	0.1	1,72382012	4,37630023
	22	0.2	2,73078764	3,59237056
	23	0.4	2,202365E-01	5,225350E-01
	24	0.5	1,998768E-03	6,159420E-01
	25	0.6	2,649501E-03	5,950950E-03
	26	0.7	**1,127300E-04**	**4,770680E-03**
	27	0.8	2,305125E-02	5,686800E-05
	28	0.9	1,772671E-03	2,410520E-04
	29	01	1,816030E-02	8,345270E-01

The influence of β, γ and population size PS (FS=PS) on optimization is reported in Table 5. In Table 5 we have compared different values on β, γ and population size PS(FS=PS) with two metrics: Mean Squared Error (MSE) and variance.

The Mean Squared Error (MSE) is measured between the BCP output and the fourth Runge-Kutta results which is referred as the first mean order and variance as the second mean order. The best results are written in bold format.

Table 5 indicates that the suggested method based on Bee Colony Programming can achieve our aim in solving differential equations. That reveals the effectiveness and accuracy of soft commuting techniques, in the same manner as conventional methods.

5. CONCLUSION

This paper focuses on the efficacy of metaheuristic methods in solving differential equations.

To solve the equilibrium profile of a symmetric suspended drop as a nonlinear differential equation, the Artificial Bee Colony Programming method has been proposed.

We have proposed the Artificial Bee Colony Programming method as a variant of bee colonies but with a symbolic representation, analysis and optimization of individuals, the representation is in tree form and the optimization and analysis is applied on symbolic expressions.

The Artificial Bee Colony Programming method solves the differential equation representing the equilibrium profile of a symmetrical suspended drop as well as the Runge-Kutta method. The results show that the efficiency of the artificial bee colony programming method is comparable to the performance of the conventional Runge-Kutta method.

However, Artificial Bee Colony Programming still has a flaw in its solution search equation, which is strong at exploration but not so much at exploitation, we investigate an hybrid approach between Artificial Bee Colony Programming and Finite Element method to improve solution.

ACKNOWLEDGMENT

Mrs. Serrat Amel wishes to thank Pr. Djebbar Bachir for his guidance, suggestions, moral support, and patience in leading her through this endeavor.

REFERENCES

[1] Hatami, M., Jing, D. (2020). Nanofluids. Academic Press, an Imprint of Elsevier.

[2] Li, Z., Zhonghua, Q., Tang, T. (2017). Numerical Solution of Differential Equations (1st ed.). Cambridge University Press.

[3] Singh, H. (2021). Advanced Numerical Methods for Differential Equations. CRC Press, Taylor & Francis Group.

[4] Zhang, Z., Karniadakis, G. (2017). Numerical Methods for Stochastic Partial Differential Equations with White Noise. Springer International Publishing.

[5] Ghorbani, H., Mahmoudi, Y., Saei, F. (2020). Numerical study of fractional Mathieu differential equation using radial basis functions. Mathematical Modelling of Engineering Problems, 7(4): 568-576. https://doi.org/10.18280/mmep.070409

[6] Radid, A., Rhofir, K. (2019). Partitioning differential transformation for solving integro-differential equations problem and application to electrical circuits. Mathematical Modelling of Engineering Problems, 6(2): 235-240. https://doi.org/10.18280/mmep.060211

[7] Goeritno, A. (2021). Ordinary differential equations models for observing the phenomena of temperature changes on a single rectangular plate fin. Mathematical Modelling of Engineering Problems, 8(1): 89-94. https://doi.org/10.18280/mmep.080111

[8] Yang, X. (2010). Nature-inspired Metaheuristic Algorithms. Luniver Press.

[9] Talbi, E. (2009). Metaheuristics. John Wiley & Sons.

[10] Yang, X. (2010). Engineering Optimization. Wiley.

[11] Karaboga, D., Ozturk, C., Karaboga, N., Gorkemli, B. (2012). Artificial bee colony programming for symbolic regression. Information Sciences, 209: 1-15. https://doi.org/10.1016/j.ins.2012.05.002

[12] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization, technical report - Tr06. Technical Report, Erciyes University.

[13] Agarwal, S., Yadav, S. (2019). A comprehensive survey on artificial bee colony algorithm as a frontier in swarm intelligence. Advances In Intelligent Systems and Computing, pp. 125-134. https://doi.org/10.1007/978-981-13-5934-7_12

[14] Chandrakala, D., Sumathi, S. (2012). Application of artificial bee colony optimization algorithm for image classification using color and texture feature similarity fusion. ISRN Artificial Intelligence, 2012: 426957. https://doi.org/10.5402/2012/426957

[15] Cuevas, E., Sención-Echauri, F., Zaldivar, D., Pérez, M. (2013). Image segmentation using artificial bee colony optimization. Handbook of Optimization, pp. 965-990. https://doi.org/10.1007/978-3-642-30504-7_38

[16] Öztürk, Ş., Ahmad, R., Akhtar, N. (2020). Variants of Artificial Bee Colony algorithm and its applications in medical image processing. Applied Soft Computing, 97: 106799. https://doi.org/10.1016/j.asoc.2020.106799

[17] Chen, J., Yu, W., Tian, J., Chen, L. (2017). Adaptive image contrast enhancement using artificial bee colony optimization. 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, pp. 3220-3224. https://doi.org/10.1109/icip.2017.8296877

[18] Akay, B., Karaboga, D. (2015). A survey on the applications of artificial bee colony in signal, image, and video processing. Signal, Image and Video Processing, 9(4): 967-990. https://doi.org/10.1007/s11760-015-0758-4

[19] Zou, W., Zhu, Y., Chen, H., Sui, X. (2010). A clustering approach using cooperative artificial bee colony algorithm. Discrete Dynamics in Nature and Society, 2010: 459796. https://doi.org/10.1155/2010/459796

[20] Ji, J., Pang, W., Zheng, Y., Wang, Z., Ma, Z. (2015). A novel artificial bee colony based clustering algorithm for categorical data. PLOS ONE, 10(5): e0127125. https://doi.org/10.1371/journal.pone.0127125

[21] Danish, Z., Shah, H., Tairan, N., Gazali, R., Badshah, A. (2019). Global artificial bee colony search algorithm for data clustering. International Journal of Swarm Intelligence Research, 10(2): 48-59. https://doi.org/10.4018/ijsir.2019040104

[22] Xu, F., Li, H., Pun, C., Hu, H., Li, Y., Song, Y., Gao, H. (2020). A new global best guided artificial bee colony algorithm with application in robot path planning. Applied Soft Computing, 88: 106037. https://doi.org/10.1016/j.asoc.2019.106037

[23] Chen, T., Sun, L. (2019). A connectivity weighting dv_hop localization algorithm using modified artificial bee colony optimization. Journal of Sensors, 2019: 1464513. https://doi.org/10.1155/2019/1464513

[24] Annepu, V., Rajesh, A. (2020). Implementation of an efficient artificial bee colony algorithm for node localization in unmanned aerial vehicle assisted wireless sensor networks. Wireless Personal Communications, 114(3): 2663-2680. https://doi.org/10.1007/s11277-020-07496-8.

[25] Chow, H., Hasan, S., Bareduan, S. (2013). Basic concept of implementing artificial bee colony (ABC) system in flow shop scheduling. Applied Mechanics and Materials, 315: 385-388. https://doi.org/10.4028/www.scientific.net/amm.315.385

[26] Muthiah, A., Rajkumar, R. (2014). A comparison of artificial bee colony algorithm and genetic algorithm to minimize the makespan for job shop scheduling. Procedia Engineering, 97, 1745-1754. https://doi.org/10.1016/j.proeng.2014.12.326

[27] Liang, Y., Chen, A., Nien, Y. (2014). Artificial bee colony for workflow scheduling. 2014 IEEE Congress On Evolutionary Computation (CEC), Beijing China, pp. 558-564. https://doi.org/10.1109/cec.2014.6900537

[28] Gao, K., Zhang, Y., Sadollah, A., Su, R. (2017). Improved artificial bee colony algorithm for solving urban traffic light scheduling problem. In 2017 IEEE Congress on Evolutionary Computation (CEC), Donostia, Spain, pp. 395-402. https://doi.org/10.1109/cec.2017.7969339

[29] Sun, L., Chen, T., Zhang, Q. (2018). An artificial bee colony algorithm with random location updating. Scientific Programming, 2018: 2767546. https://doi.org/10.1155/2018/2767546

[30] Yan, X., Chan, F., Zhang, Z., Lv, C., Li, S. (2020). A modified artificial bee colony algorithm for scheduling optimization of multi-aisle AS/RS system. Lecture Notes in Computer Science, Belgrade, Serbia, pp. 94-103. https://doi.org/10.1007/978-3-030-53956-6_9

[31] Celik, M., Karaboğa, D., Köylü, F. (2011). Artificial bee colony data miner (abc-miner). In 2011 International Symposium on Innovations in Intelligent Systems and Applications, Istanbul, Turkey, pp. 96-100. https://doi.org/10.1109/inista.2011.5946053

[32] Zorarpacı, E., Ayşe Özel, S. (2021). Privacy preserving rule-based classifier using modified artificial bee colony algorithm. Expert Systems with Applications, 183: 115437. https://doi.org/10.1016/j.eswa.2021.115437

[33] Si, T., De, A., Bhattacharjee, A. (2013). Grammatical bee colony. Swarm, Evolutionary, and Memetic Computing, Chennai, India, pp. 436-445. https://doi.org/10.1007/978-3-319-03753-0_39

[34] Koza, J.R., Rice, J.P. (1992a). Genetic Programming: The Movie. MIT Press, Cambridge, MA.

[35] Koza, J.R., Rice, J.P. (1992b). Automatic programming of robots using genetic programming. In Proceedings of Tenth National Conference on Artificial Intelligence, pp. 194-201.

[36] Boudouaoui, Y., Habbi, H., Ozturk, C., Karaboga, D. (2020). Solving differential equations with artificial bee colony programming. Soft Computing, 24(23): 17991-18007. https://doi.org/10.1007/s00500-020-05051-y

[37] Seeley, T. (1989). Social foraging in honey bees: How nectar foragers assess their colony's nutritional status. Behavioral Ecology and Sociobiology, 24(3): 181-199. https://doi.org/10.1007/bf00292101

[38] Gorkemli, B., Karaboga, D. (2019). A quick semantic artificial bee colony programming (qsABCP) for symbolic regression. Information Sciences, 502: 346-362. https://doi.org/10.1016/j.ins.2019.06.052.

[39] Xue, D. (2020). Differential Equation Solutions with MATLAB®. Berlin, Boston: De Gruyter. https://doi.org/10.1515/9783110675252

[40] Chakraverty, S., Mahato, N., Karunakar, P., Rao, T.D. (2019). Advanced Numerical and Semi-Analytical Methods for Differential Equations. John Wiley & Sons.

[41] Shi, E., Xu, C. (2021). A comparative investigation of neural networks in solving differential equations. Journal of Algorithms & Computational Technology, 15: 174830262199860.https://doi.org/10.1177/1748302621998605

[42] Han, J., Jentzen, A., Weinan, E. (2018). Solving high-dimensional partial differential equations using deep learning. Proceedings of the National Academy of Sciences, 115(34): 8505-8510.https://doi.org/10.1073/pnas.1718942115
[43] Michoski, C., Milosavljević, M., Oliver, T., Hatch, D. (2020). Solving differential equations using deep neural networks. Neurocomputing, 399: 193-212. https://doi.org/10.1016/j.neucom.2020.02.015
[44] Chen, Y.Y., Chang, Y.T., Chen, B.S. (2008). Fuzzy solutions to partial differential equations: adaptive approach. IEEE Transactions on Fuzzy Systems, 17(1): 116-127. https://doi.org/10.1109/tfuzz.2008.2005010
[45] Lu, Y., Yin, Q., Li, H., Sun, H., Yang, Y., Hou, M. (2019). The LS-SVM algorithms for boundary value problems of high-order ordinary differential equations. Advances in Difference Equations, 2019(1): 195. https://doi.org/10.1186/s13662-019-2131-3
[46] Peñuñuri, F., Péon-Escalante, R., Solís-Perales, G., Villanueva, C., Acosta, C. (2012). A differential evolutionary method for solving a class of differential equations numerically*. IFAC Proceedings Volumes, 45(12): 45-50. https://doi.org/10.3182/20120620-3-mx-3012.00058
[47] Panagant, N., Bureerat, S. (2014). Solving Partial Differential Equations Using a New Differential Evolution Algorithm. Mathematical Problems in Engineering, 2014: 747490. https://doi.org/10.1155/2014/747490
[48] Gutierrez-Navarro, D., Lopez-Aguayo, S. (2018). Solving ordinary differential equations using genetic algorithms and the Taylor series matrix method. Journal of Physics Communications, 2(11): 115010. https://doi.org/10.1088/2399-6528/aaedd2
[49] Caro, L.A.P., Mendoza, R., Mendoza, V.M.P. (2021). Application of genetic algorithm with multi-parent crossover on an inverse problem in delay differential equations. AIP Conference Proceedings, 2423(1): 020010. https://doi.org/10.1063/5.0075363
[50] Tsoulos, I., Lagaris, I. (2006). Solving differential equations with genetic programming. Genetic Programming and Evolvable Machines, 7(1): 33-54. https://doi.org/10.1007/s10710-006-7009-y
[51] Serrat, A., Djebbar, B. (2021). Approximate solution by ant colony programming to symmetrical drop suspended equilibrium equation. International Review on Modelling and Simulations (IREMOS), 14(3): 176. https://doi.org/10.15866/iremos.v14i3.19730
[52] Haynes, T. (1996). Book review: Advances in Genetic Programming edited by Kenneth E. Kinnear, Jr. (The MIT Press 1994). ACM SIGART Bulletin, 7(3): 17-18. https://doi.org/10.1145/239616.1066350
[53] Penny, D., Hendy, M., Steel, M. (1992). Progress with methods for constructing evolutionary trees. Trends in Ecology & Evolution, 7(3): 73-79. https://doi.org/10.1016/0169-5347(92)90244-6
[54] Kumar, A., Sinha, N., Bhardwaj, A. (2020). A novel fitness function in genetic programming for medical data classification. Journal Of Biomedical Informatics, 112: 103623. https://doi.org/10.1016/j.jbi.2020.103623
[55] Zhou, Y., Yang, J. (2019). Automatic design of scheduling policies for dynamic flexible job shop scheduling by multi-objective genetic programming based hyper-heuristic. Procedia CIRP, 79: 439-444. https://doi.org/10.1016/j.procir.2019.02.118

NOMENCLATURE

abs	Absolute value
Fit	Fitness value
g	Gravitational acceleration, $m.s^{-2}$
P	Probability value
V	Drop Volume, mm^3
X	Bee population

Greek symbols

α	Contact Angle, °
β	thermal expansion coefficient, K^{-1}
γ	solid volume fraction
η	Distance x
ρ	Fluid volume, $g.cm^{-3}$
σ	Interfacial tension, dyne/cm