

Shooting swarm algorithm for solving two-point boundary value problems

Suhaib Abduljabbar Altamir¹, Mohammed Abdulrazaq Kahya¹, Azzam Salahuddin Younus Aladool²

¹Department of Computer sciences, College of Education for Pure Sciences, University of Mosul, Mosul, Iraq

²Department of Mathematics, College of Education for Pure Sciences, University of Mosul, Mosul, Iraq

Article Info

Article history:

Received Jul 2, 2022

Revised Sep 14, 2022

Accepted Oct 5, 2022

Keywords:

Boundary value problems

Non-stiff problems

Shooting method

Stiff problems

Swarm algorithms

ABSTRACT

Boundary value problems (BVPs) are solved using the more detailed swarm algorithm (SA) based on particle swarm optimization (PSO) and firefly algorithm (FA). In the field of optimization techniques, both PSO and FA have good features to solve many problems in applied mathematics. Due to the sensitivity of the use of the controversial shooting method for solving BVPs, which can not able to reach the exact solution oftentimes. A shooting Swarm algorithm (SSA) is proposed based on PSO and FA. Several BVPs including stiff BVPs were principally used to investigate the SSA. The numerical experiments and analyses revealed that the algorithm was able to overcome the shooting method drawbacks. On another hand, the proposed method that is based on FA significantly reduces the number of iterations required for solving BVPs, because of its flexible properties in the exploration and exploitation phases, and it is in good agreement with the exact solution of BVPs. The SSA was investigated to solve stiff BVPs and Its efficacy has been proven with the accurate solutions.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Mohammed Abdulrazaq Kahya

Department of Computer Sciences, College of Education for Pure Sciences, University of Mosul

Al Majmoaa Street, Mosul, Iraq

Email: mohammedkahya@uomosul.edu.iq

1. INTRODUCTION

Swarm algorithms (SA) are modern meta-heuristics algorithms based on stochastic search method that are suggested to deal with complex problems inspired by the natural evolution of creatures. Due to the robust behavior and flexible nature of SA, they becomes efficient techniques to solve complex problems when used in tandem with other methods [1]. There are many effective SA inspired by the intelligent swarm behaviors of birds, whales, ants, bees, fireflies and many other types of creatures to solve numerical problems [2]-[8].

In real-world applications, in the field of applied mathematics, systems of differential equations together with a set of additional conditions represent several real world problems [7]-[10]. Such systems are called the initial value problems (IVPs), when each equation of the system can satisfy the condition at the initial point. If these equations satisfy the conditions at two different points, then called two-point boundary value problems (BVPs). However, determining numerical solutions of BVPs is difficult unless it's transformed into IVPs. Different physical phenomena cases can act at different time scales and they can occur simultaneously. Such cases are governed by BVPs and classified as stiff-BVPs. Solving this type of BVPs requires numerical integrators that produce a correct approximation of the numerical solution. One of the most effective approaches for solving BVPs numerically is the shooting method. This can be achieved by transforming the BVPs into IVPs with one free parameter. i.e. an appropriate IVP can be defined and solved by initial

value methods. The solution can be converged iteratively to the original BVPs solution. However, the iterative convergent of the solution is not certain as the shooting method depends on the gradient search method, e.g. Newton-Raphson (roots problem), which needs initial guesses that lie within the domain of convergence. This method is strongly dependent on the initial guess, which is probably far from local roots while roots must be found to obtain reasonably accurate derivative values.

In this paper, a shooting swarm algorithm (SSA) was proposed based on the particle swarm optimization (PSO) and firefly algorithm (FA) with the shooting method. The SSA offers relatively guaranteed computation compared to conventional shooting methods for solving an initial value problem. Several candidate solutions of different BVPs are analyzed showing the ability of SSA when implementing PSO and FA compared to the exact solution of the BVPs. Furthermore, the stiffness of BVPs is also investigated. The results offer a significant contribution to the field by successfully developing SSA for solving BVPs, in particular when the use of FA by providing a solution with suitable accuracy and less number of iteration. This paper's work is regulated as follows: after this introduction, section 2 will describe PSO, and FA, this is followed by boundary value problems, and the shooting method. The proposed method is detailed in section 3. Followed by the discussion of the results of this work and the main conclusion.

2. OVERVIEW

In this section, two types of SA such as PSO and FA, which are used in this work, are briefly introduced, boundary value problems, shooting method and tested boundary value problems.

2.1. Particle swarm optimization

PSO is a stochastic optimization technique inspired by the social behavior of animals like fish schooling or bird flocking presented by Kennedy and Eberhart [11]. A swarm of particles ("birds") starts with a random initial population generated (solutions) and each particle has velocity and position, which $\delta_i = \{\delta_{i1}, \delta_{i2} \dots \delta_{ik}\}, i = 1, 2, \dots, n$ represent the position vector, $v_i = \{v_{i1}, v_{i2} \dots v_{ik}\}, i = 1, 2, \dots, n$ represent the velocity vector. The positions and velocities are confined within $[\delta_{min}, \delta_{max}]$ and $[v_{min}, v_{max}]$ respectively. Each particle of the swarm is updated by two of the "best" values, "pbest" and "gbest". The individual best position obtained for the i -th particle in a swarm with k particles is called "pbest" while the best position achieves by any particle in the population is called "gbest". In general, the update equations can be interpreted as (1) and (2):

$$v_i^{t+1} = w \times v_i^t + c_1 \times rand_1 (p_i^{best} - \delta_i^t) + c_2 \times rand_2 (g_i^{best} - \delta_i^t), \quad (1)$$

$$\delta_i^{t+1} = \delta_i^t + v_i^{t+1}, \quad (2)$$

where t is the number of iterations, $w \in [0, 1]$ is the inertia weight, c_1 and c_2 are learning factors; $rand_1$ and $rand_2$ are random numbers selected between 0 and 1 [12], [13].

2.2. Firefly algorithm

FA is an optimization algorithm inspired by the behavior of fireflies beetle first proposed by Yang and Slowik [6]. There are more than two thousand firefly species that emits short flash lighting to express the important things belonging to risks, mating and sustenance [6], [14]. The firefly lighting brightness is given by the fitness function [14]. As a rule, the position of fireflies is specified by the following updating (3):

$$\delta_j^{t+1} = \delta_j^t + \beta \times \exp(-\gamma r_{ij}^2) (\delta_i^t - \delta_j^t) + \alpha (rand - 1/2), \quad (3)$$

$\alpha \in [0, 1]$ is the parameter of noise existing in the environment and β is a parameter of the initial attractiveness which is permanently set to 1. $rand$ represents a random number created from a uniform distribution in the interval $[0, 1]$. γ is the variation of the attractiveness and its value varies from 0.01 to 100 [15]-[17]. Finally, the distance between Firefly i and j is formulated as r_{ij} in the (4) as (4):

$$r_{ij} = \|(\delta_i - \delta_j)\|, \quad (4)$$

where δ_i represents the position of the i -th firefly.

2.3. Boundary value problems

Two-point BVPs happen frequently in many fields of applied sciences such as control, theoretical physics experiments and engineering branches. Often two-point BVP is analytically unsolvable, therefore resort to solving it numerically [18]-[20]. In general, two-point BVPs can be clarified in the following (5):

$$y'' = f(x, y, y'), \quad a \leq x \leq b, \quad y(a) = \alpha, \quad y(b) = \beta. \quad (5)$$

if we suppose that (5) is continuous set on D where

$$D = \{(x, y, y') : a \leq x \leq b, -\infty \leq y \leq \infty, -\infty \leq y' \leq \infty\} \quad (6)$$

and that $\frac{\partial f}{\partial y}$ and $\frac{\partial f}{\partial y'}$ are also continuous on D , then (5) has a unique solution if verified the following conditions:

- $\frac{\partial f}{\partial y}(x, y, y') > 0$ for all $(x, y, y') \in D$,
- $\left| \frac{\partial f}{\partial y}(x, y, y') \right| < M$ for all $(x, y, y') \in D$ [21]

2.4. Stiff-BVPs

In real word application, different physical phenomena performing at very different time scales which occur simultaneously. Such cases are governed by boundary value problems. The solution of these BVPs has components that decay exponentially at different speeds. Such cases are called stiff. Otherwise it calls non-stiff BVPs. The solution of such systems is normally inaccurate with the use of explicit numerical integrators. In this case, the stability is required and it can be satisfied by specifying a small size of time step [22].

2.5. Shooting method

One of the most official traditional methods to solve BVPs is the shooting method [18], [23]. The shooting method replaces the two point BVPs described in (5) with the initial value problem as (7):

$$y'' = f(x, y, y'), \quad a \leq x \leq b, \quad y(a) = \alpha, \quad y'(a) = \gamma. \quad (7)$$

where gamma is guess number. The solution to (5), denoted $y(x, \gamma)$, is an IVP solution and it satisfies the first boundary condition i.e. $y(a) = \alpha$ but typically it does not nearly close to the right boundary condition i.e. $y(b, \gamma_0) \neq \beta$. If one can determine a number gamma with:

$$y(b, \gamma) - \beta = 0 \quad (8)$$

then gamma is a root to the (8) and the IVP solution $y(x, \gamma)$ is a TPBVP solution. If the solution of (5) is unique, then $y(x, \gamma)$ is the sought TPBVP solution. Hence, any shooting method is a root seeking procedure for finding the root gamma of the (8).

The main task of the shooting method is converting the BVP into an equivalent IVP i.e finding the value of $y'(a)$. The traditional shooting method provided suitable solutions for these problems in the past epoch, but it contains some drawbacks. The basic drawback of the shooting method is that it depends on the gradient search method e.g. Newton-Raphson (roots problem) which needs to provide for initial guesses that lie within the domain of convergence. The task of obtaining perfect initial guesses is often not easy at all because it involves the use of Lagrange multipliers [24]. In the next section, an alternative method for solving BVPs is introduced by applying SA with shooting method to reduce the drawbacks of shooting method.

2.6. Tested boundary value problems

Seven boundary value problems that were obtained from the several studies, were used [25]-[28]. Table 1 summarizes a brief description of the used equations. The equations that have been exploited to test the effectiveness of proposed method.

Table 1. Tested boundary value problems

No.	Equation	Exact solution	Boundary conditions
1	$\frac{d^2y}{dx^2} = -x \frac{dy}{dx} + 3y + 4.2x$	$y(x) = x^3 + 0.9x$	$y(0) = 0, y(1) = 1.9$
2	$\frac{d^2y}{dx^2} = -\frac{y^3 - 2y^2}{2x^2}$	$y(x) = \frac{2x}{x+1}$	$y(1) = 1, y(2) = \frac{4}{3}$
3	$\frac{d^2y}{dx^2} = -\left(\frac{dy}{dx}\right)^2$	$y(x) = \ln(x)$	$y(1) = 0, y(2) = \ln(2)$
4	$\frac{d^2y}{dx^2} = \frac{2y}{x^2} - \frac{1}{x}$	$y(x) = \frac{(19x - 5x^2 - \frac{36}{x})}{38}$	$y(2) = 0, y(3) = 0$
5	$\frac{d^2y}{dx^2} = -10x \frac{dy}{dx}$	$y(x) = 1 + \frac{\operatorname{erf}\left(\frac{x}{\sqrt{0.2}}\right)}{\operatorname{erf}\left(\frac{1}{\sqrt{0.2}}\right)}$	$y(-1) = 0, y(1) = 2$
6	$\frac{d^2y}{dx^2} = \frac{-3\epsilon y}{(\epsilon + x^2)^2}$	$y(x) = \frac{x}{\sqrt{\epsilon + x^2}}$	$y(-0.1) = \frac{-0.1}{\sqrt{\epsilon + 0.01}} = -y(0.1)$
7	$\frac{d^2y}{dx^2} = \frac{-4xy' - 2y}{\epsilon + x^2}$	$y(x) = \frac{1}{x^2 + \epsilon}$	$y(-1) = y(1) = \frac{1}{1 + \epsilon}$

3. METHOD

This section is about the algorithm description of SSA based on PSO and FA with shooting, it was proposed to find the approximate solution of two-point BVPs. The detailed of the SSA computation is described in Algorithm 1.

Algorithm 1: the computation of SSA

Input: endpoints a, b , boundary conditions α, β , Tolerance TOL , Maximum number of iteration N , k =swarm size, Number of subintervals M , parameters of PSO and FA.

Output: approximations $\hat{y}(x_i)$ to exact $y(x_i)$ for each $i = 0, 1, 2 \dots M$

Step 1: $h = \frac{(b-a)}{M}$, $t = 1$,

Step 2: randomly initialize $(\hat{y}'(a))_q = \gamma_q$ $q = 1, 2 \dots k$,

Step 3: while $(t \leq N)$ do steps (4-7),

Step 4: use Rosenbrock methods [29], [30] to solve the following system on different initial conditions for $\hat{y}'(a)$;

$$y'' = f(x, y, y'), \quad (y(a))_q = \alpha, \quad (\hat{y}'(a))_q = \gamma_q. \quad q = 1, 2 \dots k \quad (9)$$

Step 5: find the best position using the proposed fitness function “ $\min |\hat{y}(b, \gamma_q) - \beta|$ ”

Step 6: if $|\text{best} - \beta| < TOL$ then for $i = 0, 1, \dots M$; set $x_i = a + ih$; output $(x_i, \hat{y}(x_i))$; the procedure is completed; stop.

Step 7: updating solutions $((\hat{y}'(a))_q = \gamma_q$ $q = 1, 2 \dots k$, the particle's position) using one of the following options: PSO option, use (1) and (2), FA option, use (3); $t = t + 1$; end while.

Step 8: output (“Maximum number of iterations exceeded”); (“The procedure was unsuccessful”); stop.

4. RESULTS AND DISCUSSION

In this section, the performance of the SSA based on PSO and FA is evaluated. This is carried out by solving different types of BVPs numerically compared to the exact solution. Furthermore, the stiffness of the numerical solution is also evaluated by choosing the different factors of stiffness. The algorithm was implemented using MATLAB (R2017b). Also, To verify the algorithm's reliability, the algorithm was re-implemented 10 times.

Several numerical problems are presented in this section to illustrate the accuracy and ability of the presented algorithm for solving two-point BVPs (see Figure 1 (Figure 1(a) to Figure 1(e)) and Table 2). Figure 1(a) to Figure 1(e) show the comparison between the exact solution and the numerical solution of BVPs for problems 1, 2, 3, 4, 5 respectively. Also, to evaluate the ability of SSA for solving stiff BVPs, two problems are considered and solved with different values of $f \epsilon$. This value controls the stiffness of the BVPs (see Figure 2 (Figure 2(a) to Figure 2(d)), Figure 3 (Figure 3(a) to Figure 3(d)), and Table 3).

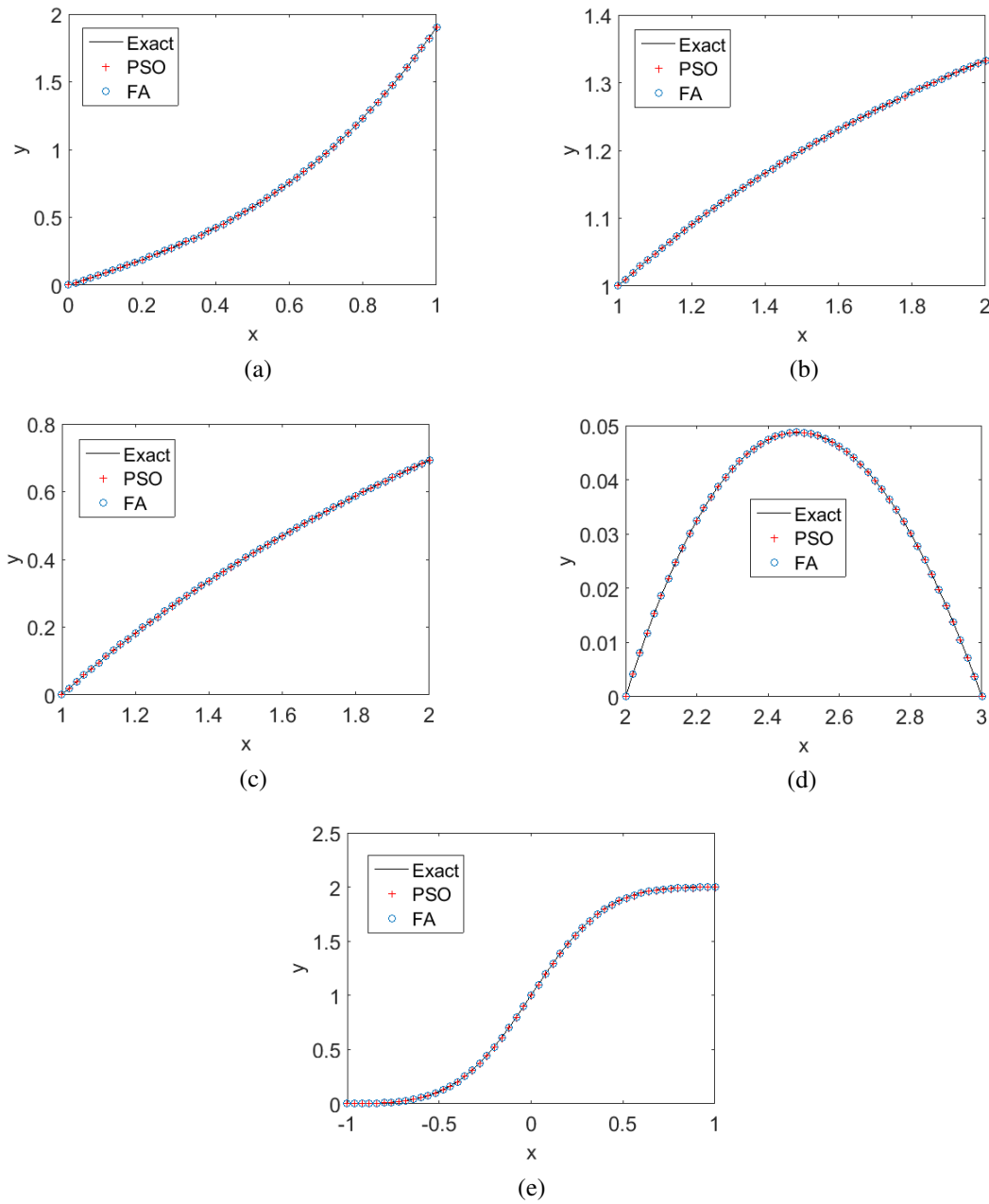


Figure 1. Computed solution using SSA of the corresponding equation of (a) problem1, (b) problem2, (c) problem3, (d) problem4, and (e) problem5 compared to exact solution of the equation itself

Table 2. numerical results of ordinary BVPs

No.	PSO		FA	
	error	#iterations	error	#iterations
1	2.154×10^{-4}	$33 \pm (3.66)$	2.154×10^{-4}	$20 \pm (1.70)$
2	1.174×10^{-4}	$31 \pm (4.71)$	8.8051×10^{-5}	$19 \pm (3.56)$
3	4.9971×10^{-5}	$29 \pm (4.24)$	4.9971×10^{-5}	$21 \pm (3.16)$
4	3.555×10^{-5}	$40 \pm (5.24)$	3.555×10^{-5}	$33 \pm (1.39)$
5	4.443×10^{-4}	$17 \pm (2.27)$	4.443×10^{-4}	$19 \pm (2.73)$

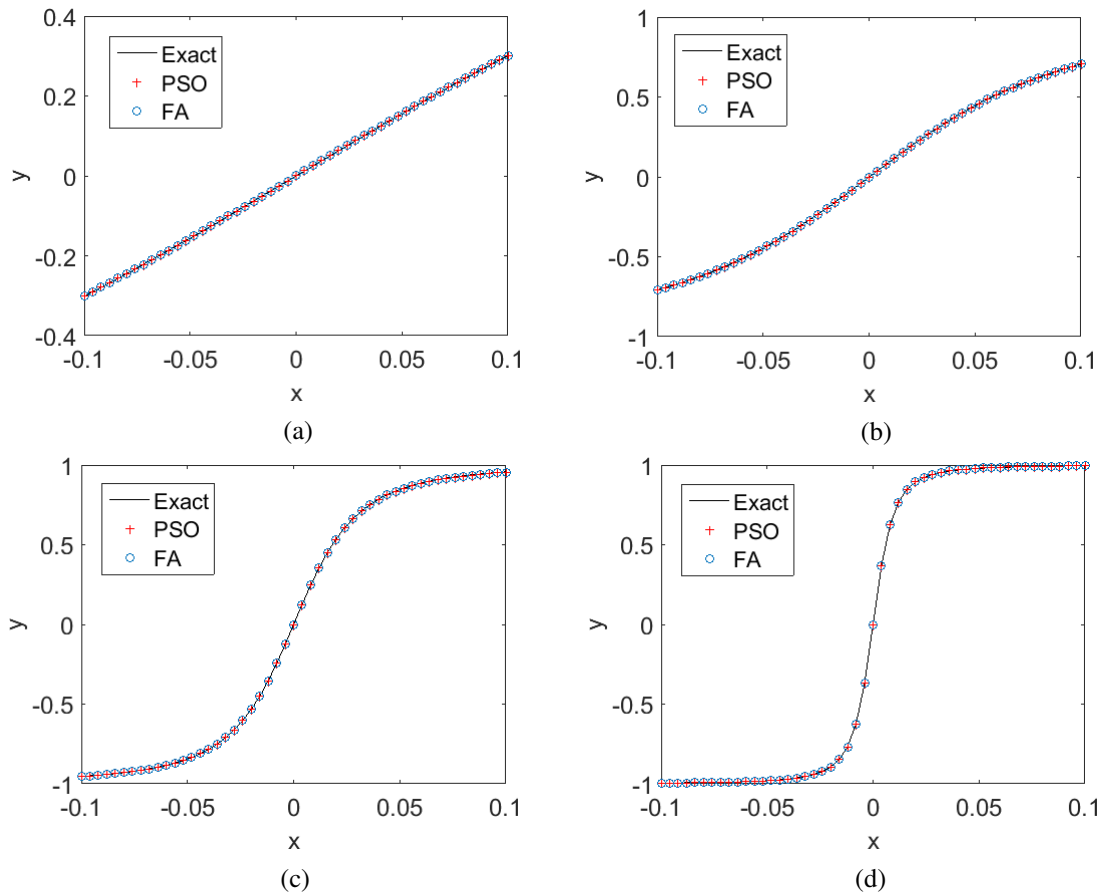


Figure 2. Computed solution using the SSA of the corresponding equation of problem 6 compared to the exact solution of the equation itself, with a different value of the stiffness factor (a) $\epsilon = 0.1$, (b) $\epsilon = 0.01$, (c) $\epsilon = 0.001$, and (d) $\epsilon = 0.0001$

Two evaluation criteria (Error and #iterations) are considered in order to verify the SSA performance as shown in the numerical solutions tables. First one is an error, which can be calculated as follows:

$$err = \max |y(x_i) - \hat{y}(x_i)|, \quad i = 0, 1, 2 \dots M \quad (10)$$

where $y(x_i)$ is the exact solution and $\hat{y}(x_i)$ is the numerical solution defined on $[a, b]$ for the corresponding example. Second, is #iterations, which represents an integer of the number of iterations on average and the value between the brackets represents the standard deviation of the iterations.

Table 3. numerical results of linear stiff BVPs

No.	PSO		#iterations	FA	
	stiffness factor ϵ	error		error	#iterations
6	0.1	8.236×10^{-5}	$24 \pm (4.33)$	8.315×10^{-5}	$17 \pm (2.94)$
	0.01	2.252×10^{-4}	$23 \pm (5.02)$	2.252×10^{-4}	$20 \pm (3.53)$
	0.001	3.331×10^{-4}	$27 \pm (2.29)$	3.331×10^{-4}	$21 \pm (2.75)$
	0.0001	1.131×10^{-3}	$27 \pm (4.17)$	1.131×10^{-3}	$19 \pm (3.30)$
7	0.1	0.0740	$19 \pm (2.70)$	0.0740	$19 \pm (2.74)$
	0.01	0.7684	$24 \pm (2.40)$	0.7684	$20 \pm (3.93)$
	0.001	6.2818	$20 \pm (3.33)$	6.2818	$22 \pm (3.02)$
	0.0001	102.1796	$22 \pm (4.23)$	102.1796	$26 \pm (3.23)$

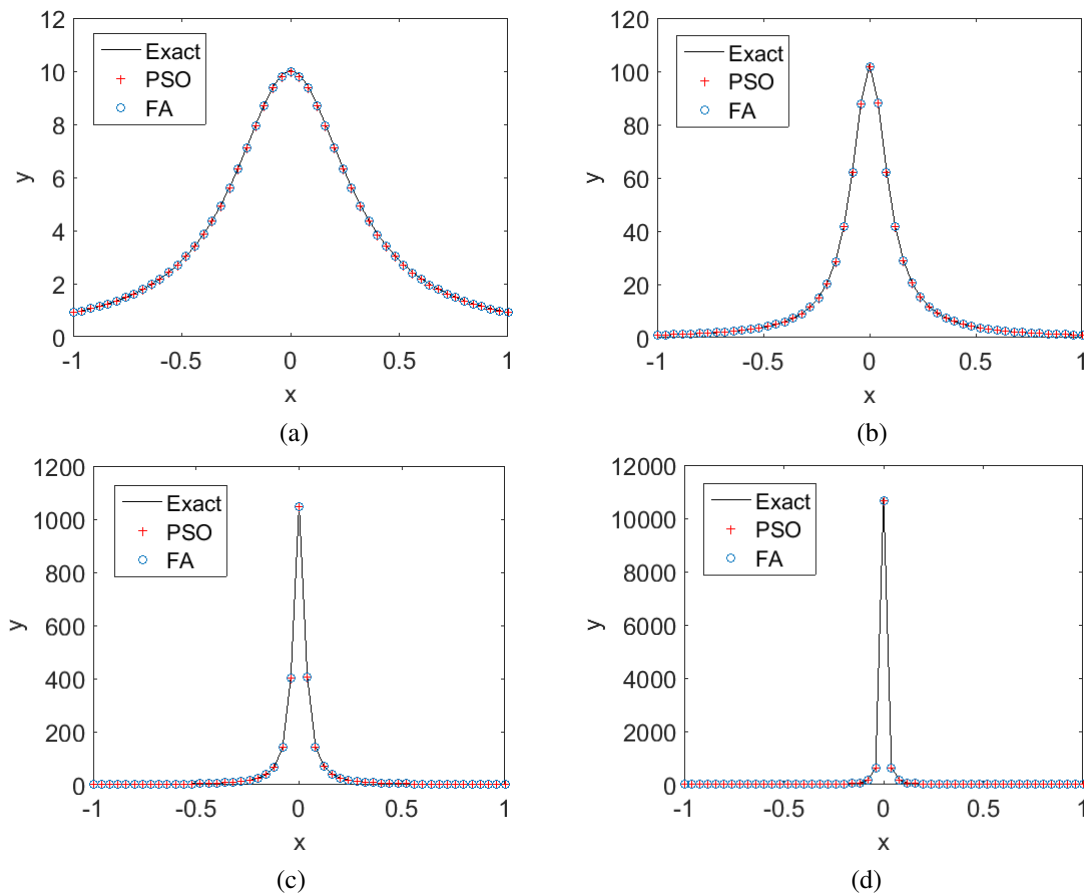


Figure 3. Computed solution using the SSA of the corresponding equation of problem 7 compared to the exact solution of the equation itself, with a different value of the stiffness factor (a) $\epsilon = 0.1$, (b) $\epsilon = 0.01$, (c) $\epsilon = 0.001$, and (d) $\epsilon = 0.0001$

The ability of swarm algorithms to solve BVPs including stiff systems and their low computational efforts permitted the development of the SSA for solving BVPs, and a comparison of numerical solutions for different examples on the given intervals and compared to their exact solution. The discussion will now take place on the comparison between the numerical and exact solution of BVPs using SSA and its dependence on the PSO and FA for finding the optimum initial conditions. Different types of two-point second-order BVPs were solved using SSA to prove the efficiency of the algorithm. The results of the first five different problems with the use of SSA showed a good agreement with a negligible error between the numerical and the exact solution of the BVPs. The results obtained from the implementation of PSO for solving BVPs are in agreement with the results obtained from the implementation of the FA with a small value of error for each problem. Further, it can be observed from the numerical solutions tables, that whether PSO or FA is implemented, has the same accuracy (i.e most of the error values are equaled for FA and PSO). This is because the same value of tolerance (TOL) is used to find the solution.

However, implementing FA in the SSA allowed faster computational time than PSO. In other words, the FA required fewer numerical iterations than the PSO to achieve a stable solution. For example, the number of iterations to solve the boundary value problem No. 1, 33 iterations through PSO while 20 iterations via FA. To elucidate the effect of the stiffness of BVPs, ϵ the value was allowed to vary, for instance ($\epsilon = 0.1$, $\epsilon = 0.01$, $\epsilon = 0.001$, $\epsilon = 0.0001$,), in the SSA, Table 3 points out this impact on the numerical solutions. Figures illustrate the ability of the SSA of solving stiff and non-stiff boundary value problems. Thus, the results showed a good agreement with a negligible error between the numerical and the exact solution of the BVPs of two stiff different examples with the use of SSA. It is also observed that the number of iterations needed for solving the stiff BVPs is significantly higher when the use of PSO is compared to the use of FA for

most values of ε . For example, the number of iterations, needed for solving problem No. 6, is ≈ 24 with the use of PSO while it is ≈ 17 with use of FA when $\varepsilon = 0.1$. The results have the same trend for most values of ε in problems No. 6 and No. 7. The consistency of the results for solving BVPs including the stiffness cases (with re-implementation of the algorithm 10 times to verify the algorithm reliability) confirms the effectiveness of the developed SSA for solving BVPs. The error value is negligible between the numerical and the exact solution of the BVPs. This may occur due to a different type of integration method used in both algorithms. However, more iterations are needed to solve the BVPs using PSO. This confirmed that the FA is a relatively low computational cost for solving BVPs than PSO. These results became useful for further study concerning swarm algorithms.

5. CONCLUSION

This work may suggest that SSA based on implementing FA for solving BVPs is more practical than implementing PSO. However, SSA remains more rigorous for solving BVPs and it can still be further developed to produce a more accurate solution for BVPs. Thus, The results showed that the SSA is remarkably effective and highlighted the critical observations that SSA permitted the comparison between the implementations of PSO and FA for solving BVPs and it's suitable to solve stiff BVPs. The scientific explanation and foremost reason for the superiority of the FA over PSO in iterations of the most boundary value problems that had tested, it has a flexible and good balance among the exploration and exploitation phases over the research stage.




REFERENCES

- [1] P. A. Vikhar, "Evolutionary algorithms: A critical review and its future prospects," *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, 2016, pp. 261-265, doi: 10.1109/ICGTSPICC.2016.7955308.
- [2] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)* 2004, pp. 1980-1987 Vol.2, doi: 10.1109/CEC.2004.1331139.
- [3] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459-471, 2007, doi: 10.1007/s10898-007-9149-x.
- [4] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646-657, 2006, doi: 10.1109/TEVC.2006.872133.
- [5] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51-67, 2016, doi: 10.1016/j.advengsoft.2016.01.008.
- [6] X.-S. Yang and A. Slowik, "Firefly algorithm," in *Swarm Intelligence Algorithms*, CRC Press, 2020, pp.163-174.
- [7] C. Xenophonos, "A parameter robust finite element method for fourth order singularly perturbed problems," *Computational Methods in Applied Mathematics*, vol. 17, no. 2, pp. 337-349, Apr. 2017, doi: 10.1515/cmam-2016-0045.
- [8] Z. Meng, Y. Zhong, G. Mao, and Y. Liang, "PSO-sono: a novel PSO variant for single-objective numerical optimization," *Information Sciences*, vol. 586, pp. 176-191, 2022, doi: 10.1016/j.ins.2021.11.076.
- [9] S. Bellew and E. O'Riordan, "A parameter robust numerical method for a system of two singularly perturbed convection-diffusion equations," *Applied Numerical Mathematics*, vol. 51, no. 2-3, pp. 171-186, 2004, doi: 10.1016/j.apnum.2004.05.006.
- [10] S. Matthews, E. O'Riordan, and G. I. Shishkin, "A numerical method for a system of singularly perturbed reaction-diffusion equations," *Journal of Computational and Applied Mathematics*, vol. 145, no. 1, pp. 151-166, 2002, doi: 10.1016/S0377-0427(01)00541-6.
- [11] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995, pp. 1942-1948 vol.4, doi: 10.1109/ICNN.1995.488968.
- [12] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *Information Processing Letters*, vol. 85, no. 6, pp. 317-325, 2003, doi: 10.1016/S0020-0190(02)00447-7.
- [13] N. A. Al-Thanoon, O. S. Qasim, and Z. Y. Algamal, "Improving nature-inspired algorithms for feature selection," *Journal of Ambient Intelligence and Humanized Computing*, vol. 13, no. 6, pp. 3025-3035, Jun. 2022, doi: 10.1007/s12652-021-03136-6.
- [14] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications. SAGA 2009. Lecture Notes in Computer Science*, O. Watanabe and T. Zeugmann, Eds. Berlin: Springer, 2009, pp. 169-178.
- [15] S. Łukasik and S. Żak, "Firefly algorithm for continuous constrained optimization tasks," in *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems. ICCCI 2009. Lecture Notes in Computer Science*, N. T. Nguyen, R. Kowalczyk, and S. Chen, Eds. Berlin: Springer, 2009, pp. 97-106.
- [16] S. K. Pal, C. Rai, and A. P. Singh, "Comparative study of firefly algorithm and particle swarm optimization for noisy non-linear optimization problems," *International Journal of Intelligent Systems and Applications*, vol. 4, no. 10, pp. 50-57, 2012, doi: 10.5815/ijisa.2012.10.06.
- [17] E. Emery, H. M. Zawbaa, K. K. A. Ghany, A. E. Hassaniien, and B. Parv, "Firefly Optimization algorithm for feature selection," *Proceedings of the 7th Balkan Conference on Informatics Conference*, 2015, pp. 1-7, doi: 10.1145/2801081.2801091.
- [18] S. N. Ha, "A nonlinear shooting method for two-point boundary value problems," *Computers Mathematics with Applications*, vol. 42, no. 10-11, pp. 1411-1420, 2001, doi: 10.1016/S0898-1221(01)00250-4.




- [19] X. Y. Li, H. L. Wang, and B. Y. Wu, "A stable and efficient technique for linear boundary value problems by applying kernel functions," *Applied Numerical Mathematics*, vol. 172, pp. 206-214, 2022, doi: 10.1016/j.apnum.2021.10.008.
- [20] W. E. Boyce, R. C. DiPrima, and D. B. Meade, *Elementary differential equations and boundary value problems*, John Wiley & Sons, 2021.
- [21] D. Greenspa and V. Casulli, *Numerical Analysis*, CRC Press, 2018.
- [22] J. E. Flaherty and R. E. O'Malley, "The numerical solution of boundary value problems for stiff differential equations," *Mathematics of Computation*, vol. 31, no. 137, pp. 66-93, 1977, doi: 10.1090/S0025-5718-1977-0657396-0.
- [23] S. C. Chapra and R. P. Canale, *Numerical methods for engineers: with programming and software application*, Boston: WCB McGraw-Hill, 1998.
- [24] Y. C. Sim, S. B. Leng, and V. Subramaniam, "A combined genetic algorithms-shooting method approach to solving optimal control problems," *International Journal of Systems Science*, vol. 31, no. 1, pp. 83-89, 2000, doi: 10.1080/002077200291488.
- [25] K. I. Ibraheem and B. M. Khalaf, "Shooting neural networks algorithm for solving boundary value problems in ODEs," *Applications and Applied Mathematics: An International Journal (AAM)*, vol. 6, no. 11, p.15, 2011.
- [26] S. Mall and S. Chakraverty, "Application of legendre neural network for solving ordinary differential equations," *Applied Soft Computing*, vol. 43, pp. 347-356, 2016, doi: 10.1016/j.asoc.2015.10.069.
- [27] L. N. M. Tawfiq and A. A. T. Hussein, "Design feed forward neural network to solve singular boundary value problems," *International Scholarly Research Notices Applied Mathematics*, vol. 2013, p. 650467, 2013, doi: 10.1155/2013/650467.
- [28] P. S. Phang, "Direct methods via multiple shooting technique for solving boundary value problems," PhD thesis, Universiti Putra, Malaysia, 2015.
- [29] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes 3rd edition: The art of scientific computing*, Cambridge university press, 2007.
- [30] L. F. Shampine and M. W. Reichelt, "The MATLAB ODE suite," *SIAM Journal on Scientific Computing*, vol. 18, no. 1, pp. 1-22, 1997, doi: 10.1137/S1064827594276424.

BIOGRAPHIES OF AUTHORS






Suhaib Abduljabbar Altamir    received Bachelor of Mathematics-Computer in University of Mosul College of Education for Pure Sciences Department of mathematics Science Mosul, Nineveh, Iraq in 1996, He received the Master degree of Numerical Analysis in University of Mosul College of Education for Pure Sciences Department of mathematics Science Mosul, Nineveh, Iraq in 2003, He received the Ph.D. of Mathematics/Computational-Computer technologies in University of Mosul College of mathematics and Computer Science Department of mathematics Science Mosul, Nineveh, Iraq in 2018, Currently He is Lecturer in Department of Computer Sciences College of Education for Pure Sciences, University of Mosul. He can be contacted at email: suhaib.altamir@uomosul.edu.iq.



Mohammed Abdulrazaq Kahya    received the Bachelor's degree in Mathematics Education from the University of Mosul in 2004, Master of Mathematics in Numerical Analysis, College of the Education for Pure Sciences University of Mosul in 2006, Iraq, and Ph.D. in Computational Mathematics, College of Computer Sciences and Mathematics University of Mosul in 2017, Iraq. Currently, he is an Assistant Professor in the Computer Sciences Department at the University of Mosul. His research interests are computational mathematics, mathematical programming, bioinformatics, and numerical analysis. He can be contacted at email: mohammedkahya@uomosul.edu.iq.



Azzam Salahuddin Younus Aladool    is a lecturer at Department of Mathematics, College of Education for Pure Science, University of Mosul, Iraq. He Holds a PhD degree in Applied Mathematics. He awarded his PhD Degree from University of Exeter, United Kingdom. His research areas are Computational Mathematics, Differential Equations, Intelligent algorithm, Mathematical Modelling. He can be contacted at email: Azzam.aladool@uomosul.edu.iq.