Engineering and Technology Journal
Journal homepage: engtechjournal.org

A Comparative Study of Various Intelligent Controllers' Performance for Systems Based on Bat Optimization Algorithm

Luay T. Rasheed [a]

[a] University of Technology, Control and Systems Engineering Department-Iraq.
60065@uotechnology.edu.iq

Submitted: 26/09/2019 Accepted: 16/01/2020 Published: 25/06/2020

K E Y W O R D S

PID controller, PIDA controller, Bat Optimization Algorithm, Higher-order systems.

A B S T R A C T

The aim of this paper is to demonstrate the performance of two intelligent controllers; the proportional-integral-derivative (PID) controller, and the proportional-integral-derivative-acceleration (PIDA) controller, based on optimization algorithm for higher order systems. In this work, bat control algorithm has been utilized to find and tune the optimal weight parameters of the controllers as simple and fast tuning technique to find the best unsaturated state and smooth control action for the systems based on the intelligent controllers. The simulation results using (Matlab Package) show that both controllers with the bat control algorithm can give excellent performance but the performance of the PIDA controller is better than that of the PID controller in terms of reducing the rising time (Tr), peak time (Tp), settling time (Ts), maximum overshoot (Mp), and steady-state error (Ess). Furthermore, the fitness evaluation value is reduced.

How to cite this article: L. T. Rasheed, "A comparative study of various intelligent controllers' performance for systems based on bat optimization algorithm," Engineering and Technology Journal, Vol. 38, No. 06, pp. 938-950, 2020.
DOI: https://doi.org/10.30684/etj.v38i6A.622

1. Introduction

In the last few decades, the PID controller has been widely used both in industry and academia in spite of several advanced control methods that have been proposed. Presently, more than 90% control systems are still PID controllers in several advanced parts of the industry, such as power systems, process control, robotics, and motion control. The majority of the controllers are still simple PID control systems because of their ease of understanding, simplicity of usage, and its effective performance [1].
The performance and stability of a PID-based control system may change drastically by the controller parameters (proportional, integral, and derivative). Today, the PID controller based on tuning control algorithms provides much convenience in engineering and therefore, several types of tuning control algorithms have been proposed for PID controllers such as Genetic Algorithm (GA) [2], Particle Swarm Optimization (PSO) [3], Ant Colony Optimization (ACO) [4], and Fruit Fly Optimization (FOA) [5].
The PIDA controller Proposed by Jung and Dorf in 1996 an extension to the conventional PID controller which has four control parameters (proportional, integral, derivative and Acceleration control parameters) and with this new term, a closed-loop system can respond faster with less

overshoot [6, 7]. Similar to the PID controllers several different tuning algorithms have been proposed for the PIDA Controllers such as Flower Pollination Algorithm (FPA) [8], Genetic Algorithm (GA) [6], Bat Optimization Algorithm (BOA) [9], and Spider Monkey Optimization (SMO) algorithm [10].
The main contribution of this work is to enhance and develop the performance of the PID controller because, in some particular situations, the PID controllers are not suitable especially for higher- order control systems. Therefore, the PIDA controller was introduced in 1996 and utilized to deliver faster and smoother response. In addition, it is more suitable for the higher-order plants compared to the PID controller. This is done by adding an additional zero to the standard PID structure to derive the PIDA structure of the controller. The addition of an extra zero to the PID controller will change the root locus of the third-order plant in order to make dominant roots more dominant by eliminating the effects of non-dominant roots [6, 7].
The remainder of this paper is organized with five sections: Section two describes the design of the intelligent PID and PIDA controllers. Section three explains in details the Bat optimization algorithm. In section four, the performance of the intelligent PID and PIDA controllers are presented through the simulation results and discussion. Finally, the conclusions are explained in section five.

2. Intelligent Controllers Design

A PID controller is commonly employed in several industrial applications because of its ability to enhance the dynamic behavior as well as to diminish or eliminate the steady-state error. This is because the derivative part adds a finite zero to the open loop transfer function and enhances the transient response and the integral part adds a pole at the origin, thus increasing system type by one and diminishing the steady state error due to a step function to zero. Besides, it is easy to use, reliable with strong adaptation performance, and its gains can be separately and simply adjusted [11].
The block diagram of this intelligent controller which consists of a PID controller and BAT algorithm is shown in Figure 1. The BAT algorithm has a strong adaptation performance, high dynamic characteristics, and good robustness performance because of its ability to find and tune the PID control parameters.

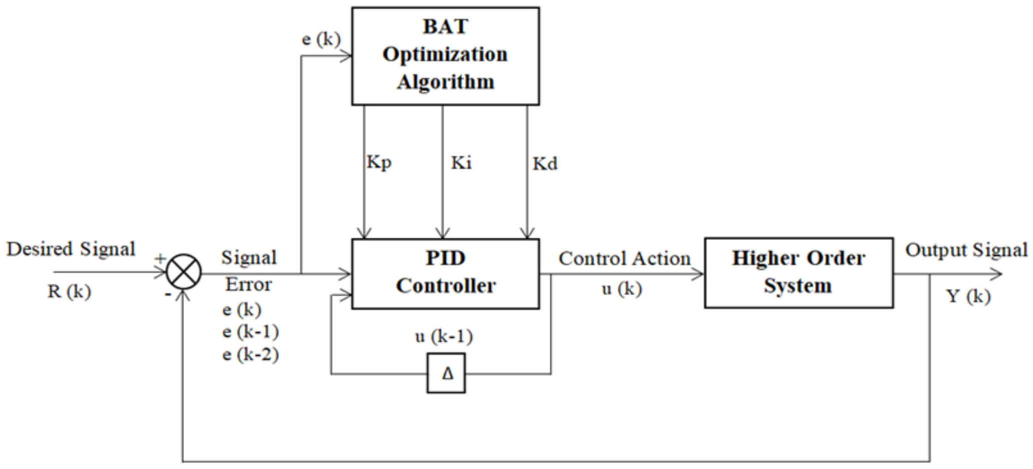

Figure 1: The block diagram of the intelligent PID controller for the higher-order system.

The equation of the PID controller in the discrete-time can be obtained by taking the first derivative of Eq. (1) as follows:

$$\dot{u}(t) = kp\,\dot{e}(t) + ki\,e(t) + kd\,\ddot{e}(t) \quad (3)$$

Applying the Backward difference formula to $\dot{u}(t)$, $\dot{e}(t)$ and $\ddot{e}(t)$ as follows:

$$\dot{u}(t) = \frac{u(k)-u(k-1)}{ts} \quad (4)$$

$$\dot{e}(t) = \frac{e(k)-e(k-1)}{ts} \quad (5)$$

$$\ddot{e}(t) = \frac{\dot{e}(k)-\dot{e}(k-1)}{ts} \quad (6)$$

Where, ts is the sampling time.
Substitute Eqs. (4, 5, and 6) in Eq. (3) gives Eq. (7) as follows:

$$\frac{u(k)-u(k-1)}{ts} = kp\,\frac{e(k)-e(k-1)}{ts} + ki\,e(k) + kd\,\frac{\dot{e}(k)-\dot{e}(k-1)}{ts} \qquad (7)$$

The PID controller's output in the time domain and the transfer function (Laplace Transform) of such a controller can be described in Eqs. (1 and 2) [12, 13]:

$$u(t) = kp\,e(t) + ki\int_0^t e(t)\,dt + kd\,\frac{de(t)}{dt} \qquad (1)$$

$$G_c(s) = \frac{U(s)}{E(s)} = kp + ki\,S^{-1} + kd\,S \qquad (2)$$

Where, kp is the proportional gain, ki is the integration gain, kd is the differentiation gain, $e(t)$ is the error signal and $u(t)$ is the control action.
Applying the Backward difference formula on $\dot{e}(k)$ and $\dot{e}(k-1)$ in Eq. (7) gives Eq. (8) as follows:

$$\frac{u(k)-u(k-1)}{ts} = kp\,\frac{e(k)-e(k-1)}{ts} + ki\,e(k) + kd\,\frac{\frac{e(k)-e(k-1)}{ts}-\frac{e(k-1)-e(k-2)}{ts}}{ts} \qquad (8)$$

Solving for $u(k)$ finally the equation of the PID controller in the discrete-time is written as follows:

$$u(k) = u(k-1) + Kp\,\big(e(k)-e(k-1)\big) + Ki\,e(k) + Kd\,(e(k) - 2e(k-1) + e(k-2)) \qquad (9)$$

Where, $Kp = kp$, $Ki = ki * ts$, and $Kd = \frac{kd}{ts}$

The PID controller structure based on Eq. (9) for controlling the higher order systems is shown in Figure 2.

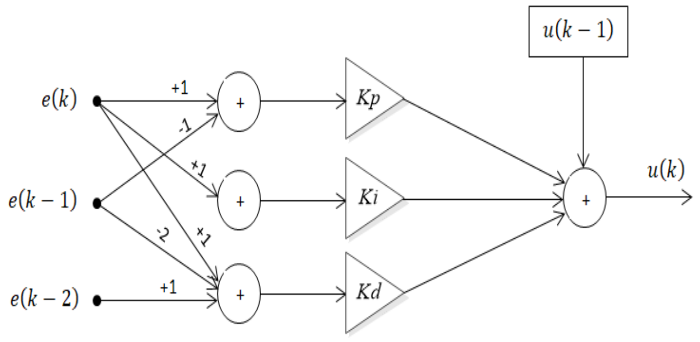

Figure 2: The structure of the PID controller

The block diagram of the PIDA intelligent controller which composed of a PIDA controller and bat algorithm is depicted in Figure 3.

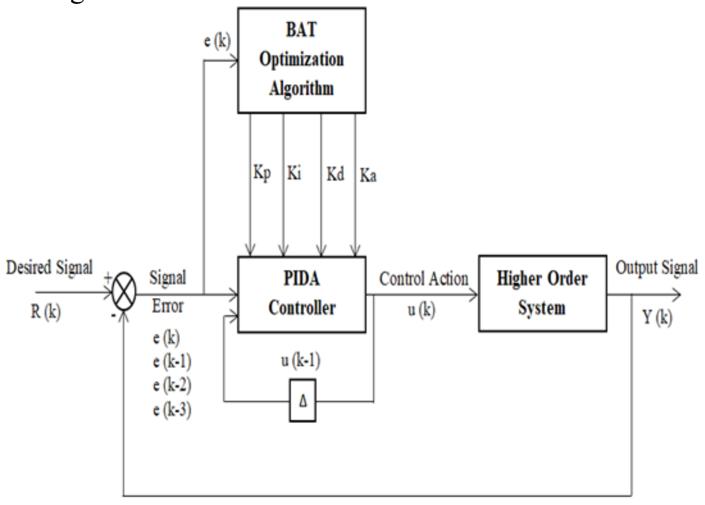

Figure 3: The block diagram of the intelligent PIDA controller for the higher-order system.

$$G_c(s) = \frac{U(s)}{E(s)} = kp + ki\,S^{-1} + \frac{kd\,S}{(S+d)} + \frac{ka\,S^2}{(S+d)(S+e)} \qquad (11)$$

$$G_c(s) = \frac{U(s)}{E(s)} = \frac{K(S+a)(S+b)(S+c)}{S(S+d)(S+e)} \quad (12)$$

According to equation (12), the PIDA controller has three zeros and three poles. Once (a,b,c) << (d,e), the two poles at $s = -d$ and $s = -e$ are neglected in design but this additional zero changes the root locus of the third-order system by making dominant roots more dominant. Due to this, the PIDA transfer function in Eqs. (11 and 12) can be rewritten as follows:

$$G_c(s) = \frac{U(s)}{E(s)} = kp + ki\, S^{-1} + kd\, S + ka\, S^2 \quad (13)$$

$$G_c(s) = \frac{U(s)}{E(s)} = \frac{ka\, S^3 + kd\, S^2 + kp\, S + ki}{S} \quad (14)$$

The equation of the PIDA controller in the discrete-time can be obtained by taking the first derivative of Eq. (10) as in the following:

$$\dot{u}(t) = kp\, \dot{e}(t) + ki\, e(t) + kd\, \ddot{e}(t) + ka\, \dddot{e}(t) \quad (15)$$

The output of the PIDA controller in the time domain and its Laplace Transform (transfer function) are symbolized in Eqs. (10, 11, and 12) [8]:

$$u(t) = kp\, e(t) + ki \int_0^t e(t)\, dt + kd\, \frac{de(t)}{dt} + ka \frac{d^2 e(t)}{dt^2} \quad (10)$$

Applying the Backward difference formula to $\dot{u}(t)$, $\dot{e}(t)$,$\ddot{e}(t)$, and $\dddot{e}(t)$ as in Eqs. (4, 5, 6, and 16) the Eq. (15) is rewritten as in Eq. (17):

$$\dddot{e}(t) = \frac{\ddot{e}(k) - \ddot{e}(k-1)}{ts} \quad (16)$$

$$\frac{u(k)-u(k-1)}{ts} = kp\, \frac{e(k)-e(k-1)}{ts} + ki\, e(k) + kd\, \frac{\dot{e}(k)-\dot{e}(k-1)}{ts} + ka\, \frac{\ddot{e}(k)-\ddot{e}(k-1)}{ts} \quad (17)$$

Applying the Backward difference formula on $\dot{e}(k)$, $\dot{e}(k-1)$, $\ddot{e}(k)$ and $\ddot{e}(k-1)$ in Eq. (17) gives Eq. (18) as follows:

$$\frac{u(k)-u(k-1)}{ts} = kp\, \frac{e(k)-e(k-1)}{ts} + ki\, e(k) + kd\, \frac{\frac{e(k)-e(k-1)}{ts} - \frac{e(k-1)-e(k-2)}{ts}}{ts} + ka\, \frac{\frac{\dot{e}(k)-\dot{e}(k-1)}{ts} - \frac{\dot{e}(k-1)-\dot{e}(k-2)}{ts}}{ts} \quad (18)$$

Applying the Backward difference formula once again on $\dot{e}(k)$, $\dot{e}(k-1)$ in Eq. (18) with some rearrangement and Solving for $u(k)$ finally gives the discrete-time PIDA controller as in Eq. (19):

$$u(k) = u(k-1) + Kp\left(e(k) - e(k-1)\right) + Ki\, e(k) + Kd\,(e(k) - 2e(k-1) + e(k-2)) + Ka\,(e(k) - 3e(k-1) + 3e(k-2) - e(k-3)) \quad (19)$$

Where, $Kp = kp$, $Ki = ki * ts$, $Kd = \frac{kd}{ts}$, and $Ka = \frac{ka}{ts^2}$

The PIDA controller structure based on Eq. (19) for controlling the higher-order systems is illustrated in Figure 4.

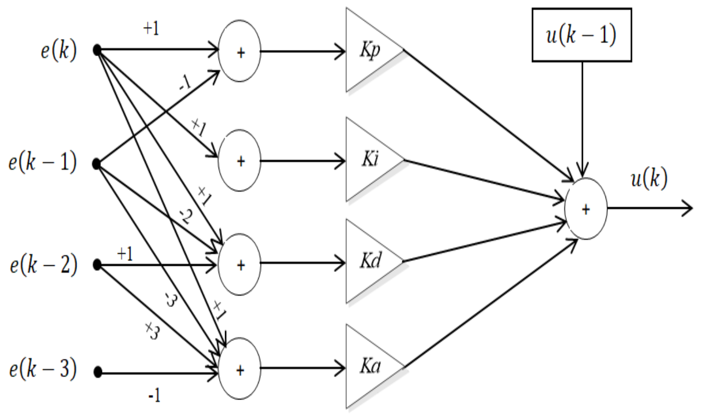

Figure 4: The structure of the PIDA controller

3.BAT Optimization Algorithm

Bat algorithm is a relatively new meta-heuristic swarm algorithm for global optimization and it is introduced and developed in 2010 by Xin-She Yang. This algorithm is inspired by the echolocation behavior of the microbats, with varying pulse rates of emission and loudness [14].

The bat optimization algorithm can be explained in three main rules: The first rule is estimating the optimal distance to the food using the echolocation phenomena. In the Second rule, the bats fly randomly in the search space with a certain velocity at a certain position with a fixed frequency. However, the wavelength and bat loudness can vary according to their distance between food and the entire bat current position. Finally, the third rule followed by bat algorithm is linearly decreasing the behavior of bat loudness factor [18]. The steps of bat optimization algorithm are given as follows [15, 16]:

Step 1: Initialize the algorithm parameters such as dimension of the problem (d), population size (N), maximum number of iterations $(Iter)$, minimum frequency (f_{min}), maximum frequency (f_{max}), loudness of bats (A), the initial Pulse emission rate (ro), pulse emission rate of bats (Υ), and the initial velocity of bats (v).

Step 2: In BA, each bat position represents a feasible solution and the initial population for each bat can be generated randomly as in Eq. (20):

$$x_{ij} = x_j^l + rand() * (x_j^u - x_j^l) \qquad (20)$$

Where; $i = 1, \dots, N,\ j = 1, \dots, d,\ x_j^l$ is the lower boundary of the dimension j and equals to 0, and x_j^u is the upper boundary of dimension j and equals to 10.

Step 3: Evaluate the cost function (MSE) as in Eq. (21) for each bat and store the results in $(cost)$ vector.

$$MSE(i) = \frac{1}{N}\sum_{i=1}^{N}(R - Y\ )^2 \qquad (21)$$

Where; R is the desired signal, Y is the output signal, N is the population size.

Step 4: Find the best bat position x^* which is the bat that has the smallest (MSE) value among all bats in $(cost)$ vector and consider its (MSE) value as (f_{min}).

Step 5: Update the frequency, velocity, and position for the i^{th} bat as shown in Eqs. (22, 23, and 24) respectively.

$$f_{i=}f_{min} + (f_{max} - f_{min}) * rand() \qquad (22)$$

$$v_i^{ii} = v_i^{ii-1} + \left(x_i^{ii-1} - x^*\right) * f_i \qquad (23)$$

$$x_i^{ii} = x_i^{ii-1} + v_i^{ii} \qquad (24)$$

Where; $ii = 1, \dots, Iter.$

Step 6: A local search is carried out if the rate of pulse emission by the i^{th} bat (Υ_i) is less than a randomly generated number, a new solution is generated for the i^{th} bat via a random walk to improve the variability of the possible solutions as in Eq. (25)

$$x_i^{new} = x^* + \epsilon * < A >^{ii} \qquad (25)$$

Where; ϵ is a scaling factor generated randomly of interval [-1, 1], and $<A>^{ii}$ is the average loudness of all bats at ii iteration.

Step 7: Evaluate the cost function value (MSE_i) for the i^{th} bat as in equation (21) and store the result in ($costn$) variable.

Step 8: For the i^{th} bat, if its new cost function value ($costn_i$) is smaller than its previous cost function value and its loudness (A_i) is bigger than a randomly generated number then the new values of its cost function, loudness and Pulse rate are calculated as in Eqs. (26, 27, and 28) as follows:

$$cost_i = costn \qquad (26)$$

$$A_i = \alpha * A_i \qquad (27)$$

$$\Upsilon_i = ro * (1 - exp(-\beta * ii)) \qquad (28)$$

Where; α and β are constants between [0, 1].

Step 9: For the i^{th} bat, if its ($costn_i$) is smaller than f_{min} then the best bat position (x^*) and minimum frequency (f_{min}) can be updated according to the Eqs. (29 and 30) as follows:

$$x^* = x_{ij} \qquad (29)$$

$$f_{min} = costn_i \qquad (30)$$

Step 10: If the maximum number of bats (N) is reached go to Step 11. Otherwise, go to Step 5.

Step 11: Stop if the maximum number of iterations ($Iter$) is reached. Otherwise, Step 5 to Step 11 is repeated.

The typical flow chart of bat algorithm for the PID and PIDA controllers is shown in Figure 5.

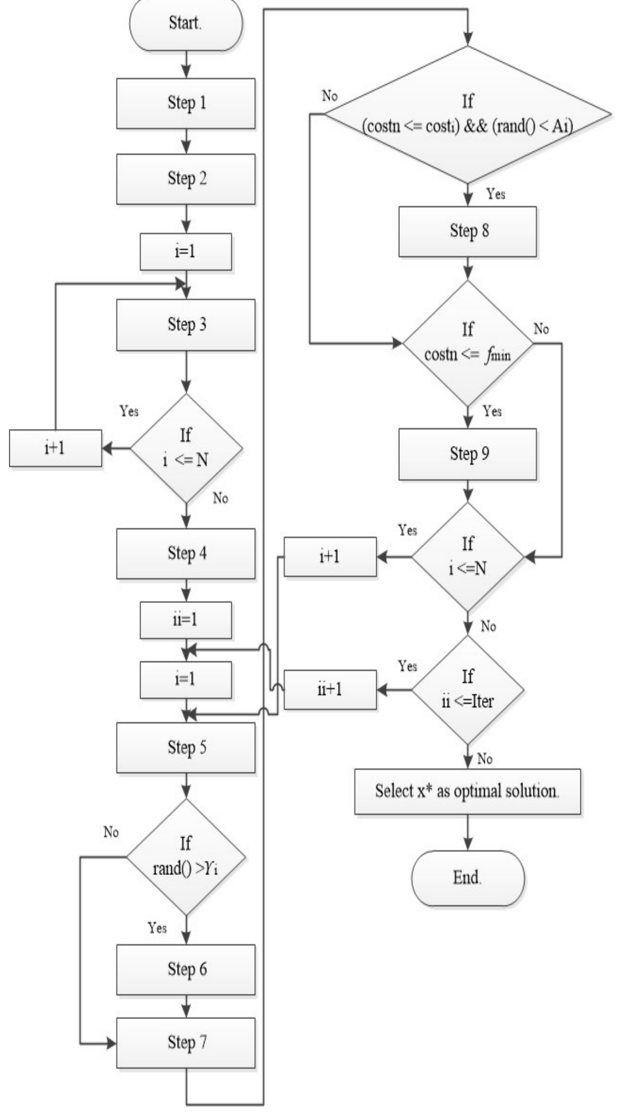

Figure 5: The flowchart of the bat optimization algorithm

4. Simulation Results and Discussions

The simulation results were obtained using Matlab package to show the performance of the two intelligent controllers. The performance of the PIDA controller based on the bat control algorithm is compared with that of the PID controller based on the same control algorithm. The parameters of the bat algorithm are: Population size is 25, Maximum iteration number is 30, Dimension of the problem is 3 for PID controller and 4 for PIDA controller, α and β are 0.5, Initial Pulse emission rate is 0.001, Minimum frequency is 0, Maximum frequency is 0.01, and the loudness is selected as a random number. The comparison has been done through the dynamic behavior of the two higher-order systems as follows:

Higher-Order System 1

The first higher-order system has been taken from [11] and can be symbolized as follows:

$$G(s) = \frac{400}{S(S^2+30S+200)} \qquad (31)$$

The Matlab simulation is carried out for the PID and PIDA controllers based on the bat control algorithm as shown in Figures 1 and 3, respectively. The bat tuning control algorithm is used off-line in the simulation in order to follow a desired signal for the higher-order system. By applying the Shannon theorem, the sampling time (ts) equals to $0.0133\,sec$ based on the time constant of the system which equals to $\tau = 0.0667$ depending on natural frequency $wn = 14.1421\ rad/sec$ and the damping ratio $\zeta = 1.0607$ of the system.

Figure 6 shows the unit step change open loop response for the system which has unstable response.

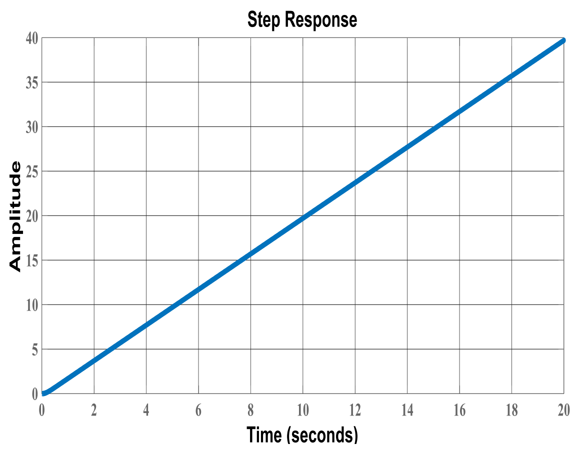

Figure 6: The open loop response of the higher-order system 1.

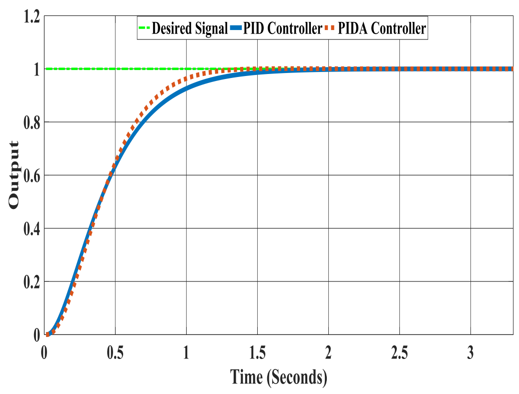

Figure 7: The output responses of the intelligent controllers for the higher-order system 1.

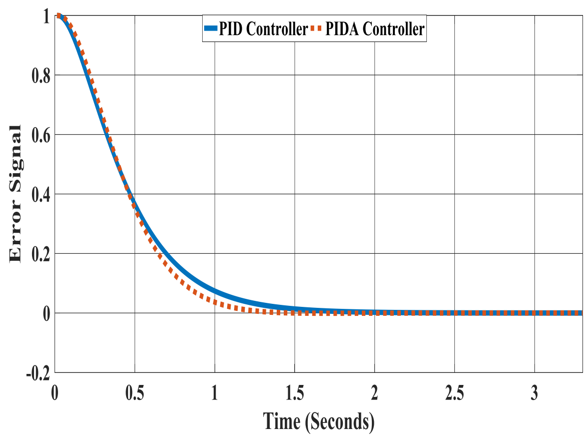

Figure 8: The error signals of the intelligent controllers for the higher-order system 1

The optimal tuning parameters and the dynamic higher-order system behavior such as rising time, peak time, settling time, overshoots, and steady state error after applying both intelligent controllers are depicted in Figure 7 and Table 1.

Table 1: Sets of the intelligent controllers' parameters with the output response for system 1.

Parameter	PID	PIDA
Kp	0.4483	0.7793
Ki	0.0013	0.0016
Kd	1.5960	0.1558
Ka	-	0.5821
Tr	0.7644 Sec	0.6445 Sec
Tp	0 Sec	0 Sec
Ts* *2% error	1.3800 Sec	1.100 Sec
Mp %	0	0
Ess	0	0

In Figure 8, the error signals of both closed-loop controllers system were a small value in the transient response and it has become zero at the steady-state response. The action responses of both intelligent controllers were smooth without oscillation response and no spikes behavior occurs as shown in Figure 9.

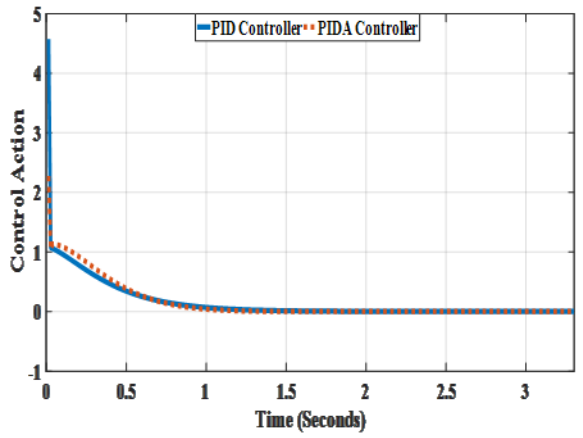

Figure 9: The control actions of the intelligent controllers for the higher-order system 1.

Figure 10 clearly shows the improved performance indices of the intelligent controllers based on the Mean Square Error.

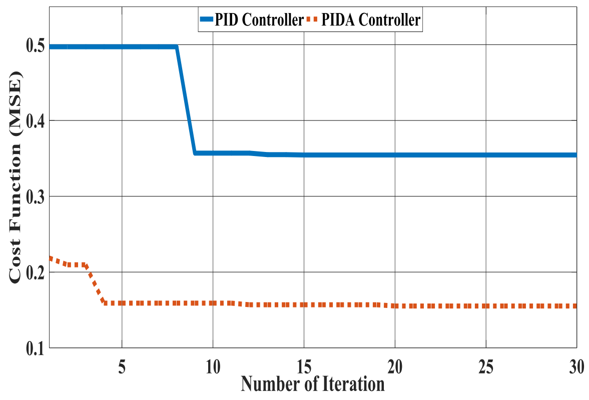

Figure 10: The performance indices (MSE) of the intelligent controllers for the higher-order system 1.

Higher-Order System 2

The second higher-order system has been taken from [8] and can be expressed as follows:

$$G(s) = \frac{0.15}{S^2+1.1S+0.2} e^{-1.5S} \qquad (32)$$

$$e^{-1.5S} = \frac{1-0.75S}{1+0.75S} \qquad (33)$$

$$G(s) = \frac{-0.1125S+0.15}{0.75S^3+1.825S^2+1.25S+0.2} \qquad (34)$$

The Matlab simulation is carried out for the PID and PIDA controllers based on the bat control algorithm as shown in Figures 1 and 3, respectively. The bat tuning control algorithm is used off-line in the simulation in order to follow a desired signal for the higher-order system. By applying the Shannon theorem, the sampling time (ts) equals to $0.1815\, sec$ based on the time constant of the system which equals to $\tau = 0.9077$ depending on natural frequency $wn = 1.0774\, rad/sec$ and the damping ratio $\zeta = 1.0225$ of the system.

The unit step change open loop response for the system is shown in Figure 11 which reveals that the system is stable.

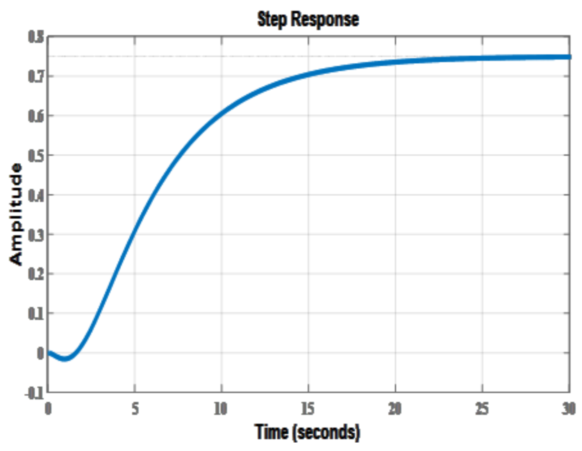

Figure 11: The open loop response of the higher-order system 2.

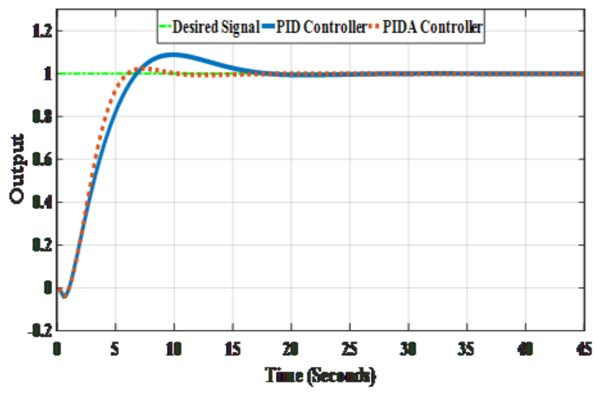

Figure 12: The output responses of the intelligent controllers for the higher-order system 2.

The optimal tuning parameters and the dynamic higher-order system behavior such as rising time, peak time, settling time, overshoots, and steady state error after applying both intelligent controllers are depicted in Figure 12 and Table 2.

Table 2: Sets of the intelligent controllers' parameters with the output response for system 2

Parameter	PID	PIDA
Kp	1.0633	1.4915
Ki	0.0786	0.0780
Kd	3.8904	2.4550
Ka	-	1.3517
Tr	6.7840 Sec	6.0325 Sec
Tp	9.8895 Sec	7.4415 Sec
Ts *2% error*	16.3084 Sec	8.4662 Sec
Mp %	8.8425	2.3670
Ess	0	0

The error signals of both closed-loop controllers system were a small value in the transient response and it has become zero at the steady-state response as shown in Figure 13.

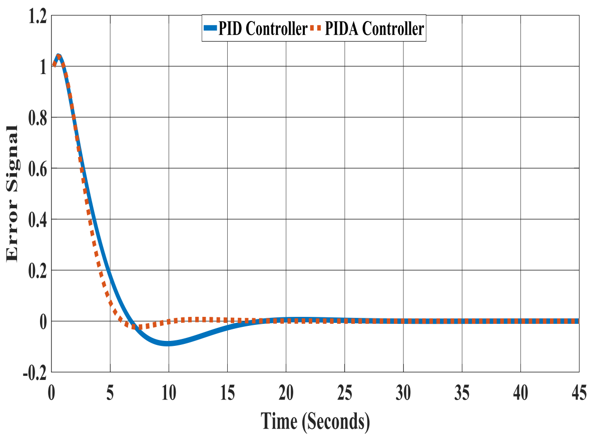

Figure 13: The error signals of the intelligent controllers for the higher-order system 2.

Figure 14 shows the action responses of both intelligent controllers were smooth without oscillation response and no spikes behavior occurs.

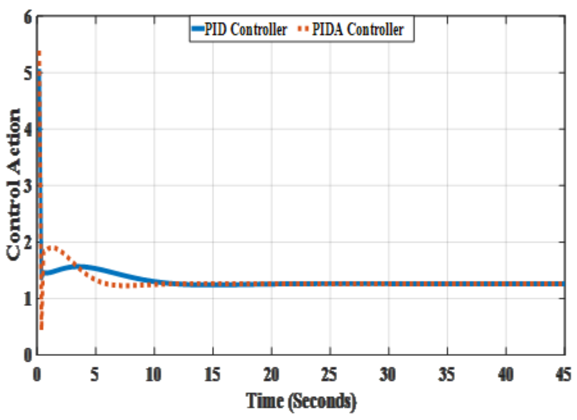

Figure 14: The control actions of the intelligent controllers for the higher-order system 2.

Figure 15 clearly shows the improved performance indices of the intelligent controllers based on the Mean Square Error.

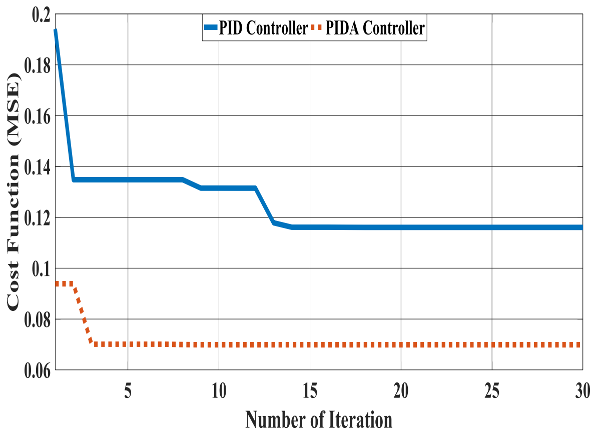

Figure 15: The performance indices (MSE) of the intelligent controllers for the higher-order system 2

5. Conclusion

The numerical simulation results of the PID and PIDA controllers based on bat optimization algorithm have been presented in this paper for the higher-order systems. The bat tuning control algorithm shows the following capabilities:

1. The off-line bat control algorithm has the ability of fast finding and tuning the optimal parameters of the controller with the minimum fitness evaluation number.
2. A proper control action was obtained as a smooth without oscillation response and no spikes behavior occurs.
3. The numerical simulation results for both controllers based on bat tuning control algorithm show that both controllers can give excellent performance but the performance of the PIDA controller for higher order systems is better than that of the PID controller in terms of reducing the rising time, peak time, settling time, overshoot and steady-state error. Moreover, the fitness evaluation number is reduced.

References

[1] E. Sariyildiz, H. Yu, and K. Ohnishi, “A practical tuning method for the robust PID controller with velocity feed-back,” *Multidisciplinary Digital Publishing Institute (MDPI) machines*, Vol. 3, No. 3, pp. 208-222, August 2015.

[2] D. C. Meena, and A. Devanshu, “Genetic algorithm tuned PID controller for process control,” International Conference of IEEE on Inventive Systems and Control (ICISC), Coimbatore, India, pp. 1-6, August, 19-20 January 2017.

[3] M. I. Solihin, L. F. Tack, and M. L. Kean, “Tuning of PID controller using particle swarm optimization (PSO),” Proceeding of the International Conference on Advanced Science, Engineering and Information Technology, Hotel Equatorial Bangi-Putrajaya, Malaysia, pp. 458-461, 14 - 15 January 2011.

[4] I. Chiha, N. Liouane, and P. Borne, “Tuning PID controller using multiobjective ant colony optimization,” *Hindawi Publishing Corporation Applied Computational Intelligence and Soft Computing*, Vol. 2012, Article ID 536326, pp. 1-7, 2012.

[5] J. Han, P. Wang, and X. Yang, “Tuning of PID controller based on fruit fly optimization algorithm,” Proceedings of IEEE International Conference on Mechatronics and Automation, Chengdu, China, pp. 409-413, 5 – 8 August 2012.

[6] S. Sornmuang, and S. Sujitjorn, “GA-based optimal PIDA controller design,” ISTASC'10 Proceedings of the 10th WSEAS international conference on Systems theory and scientific computation, Taipei, Taiwan, pp. 192-197, 20 – 22 August, 2010.

[7] D. K. Sambariya, and D. Paliwal, “Comparative design and analysis of PIDA controller using Kitti's and jung-dorf approach for third order practical systems,” *British Journal of Mathematics & Computer Science*, Vol. 16, No. 5, pp. 1-16, 2016.

[8] N. Pringsakul, C. Thammarat, S. Hlangnamthip, and D. Puangdownreong, “Obtaining optimal PIDA controller for temperature control of electric furnace system via flower pollination algorithm,” *WSEAS Transactions on Systems and Control*, Vol. 14, pp. 1-7, 2019.

[9] D. K. Sambariya, and D. Paliwal, “Design of PIDA controller using bat algorithm for AVR power system,” *Advances in Energy and Power*, Vol. 4, No. 1, pp. 1-6, 2016.

[10] A. Sharma, H. Sharma, A. Bhargava, and N. Sharma, “Optimal design of PIDA controller for induction motor using spider monkey optimization algorithm,” *International Journal of Metaheuristics*, Vol. 5, Nos. 3/4, pp. 278-290, 2016.

[11] K. E. Dagher, “Design of an Auto-Tuning PID controller for systems based on slice genetic algorithm,” *Iraqi Journal of Computers, Communications and Control & Systems Engineering (IJCCCE)*, Vol. 13, No. 3, pp. 1-9, 2013.

[12] X. Liu, C. Xu, C. Huang, and W. Shiand, “The research of tobacco leaf roasting control tactics based on fuzzy PID, ” *Advances in Intelligent and Soft Computing*, Vol. 1, Springer, pp. 185-190, 2011.

[13] M. S. Saad, H. Jamaluddin and I. Z. M. Darus, “PID controller tuning using evolutionary algorithms,” *WSEAS Transactions on Systems and Control*, Vol. 7, Issue 4, pp. 139-149, October 2012.

[14] X-S Yang, and X. He, “Bat algorithm: literature review and applications,” *International Journal of Bio-Inspired Computation*, Vol. 5, No. 3, pp. 141-149, 2013.

[15] N. A. Nor'Azlan, N. A. Selamat, and N. M. Yahya, "Multivariable PID controller design tuning using bat algorithm for activated sludge process," IOP Conference Series: Materials Science and Engineering , Vol. 342, pp. 1-9, 2018.

[16] D. K. Sambariya, and D. Paliwal, "Design of PIDA controller using bat algorithm for AVR power system," *Advances in Energy and Power*, Vol. 4, No. 1, pp. 1-6, 2016.