Evolving Systems (2024) 15:939–964
https://doi.org/10.1007/s12530-023-09526-9

ORIGINAL PAPER

Hunger games pattern search with elite opposite-based solution for solving complex engineering design problems

Serdar Ekinci[1] · Davut Izci[1,2] · Erdal Eker[3] · Laith Abualigah[4] · Cuong-Le Thanh[4] · Samir Khatir[4,5]

Received: 13 March 2023 / Accepted: 14 July 2023 / Published online: 29 July 2023
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

Abstract

The hunger games search (HGS) algorithm is designed to tackle optimization problems, however, issues such as local minimum stagnation and immature convergence hinder its effectiveness. To address these limitations, this study introduces a novel improved HGS (Imp-HGS) algorithm. The Imp-HGS algorithm uses pattern search (PS) and elite opposition-based learning (OBL) mechanisms to enhance exploitation and exploration, respectively. The algorithm's performance is tested across the CEC2019 and CECE2020 test suites along with three different engineering design problems, including identifying an infinite impulse response (IIR) model, training a multilayer perceptron (MLP), and designing a proportional-integral-derivative (PID) controller for a doubly fed induction generator (DFIG)-based wind turbine system. The test functions demonstrated superior performance of the Imp-HGS algorithm over a wide range of state-of-the-art algorithms. The ablation tests using CEC2020 test suite also demonstrate the wisely integration of the PS and elite OBL mechanisms as significant improvements are achieved. The statistical results demonstrate the significance of Imp-HGS in the IIR system identification as it consistently achieved lower average errors, lower standard deviations, competitive best results, and satisfactory worst results compared to the other algorithms. Moreover, the Imp-HGS algorithm consistently demonstrates better performance in terms of average classification rates across various datasets, showcasing its effectiveness in solving classification problems, making it a good tool for MLP training. Lastly, the Imp-HGS algorithm's ability to eliminate overshoot, achieve faster rise time, shorter settling time, and minimal peak time showcases its effectiveness in achieving stable and efficient operation of the wind turbine system. The computational times also confirm the efficacy of the Imp-HGS algorithm for all considered real-world engineering problems. Overall, the results show that the proposed algorithm outperformed other competitive approaches, cementing its status as a highly promising tool for tackling a wide range of complex engineering optimization problems.

Keywords Hunger games search · Elite opposition-based learning · Pattern search · Optimization · Engineering design problems

1 Introduction

1.1 Background

Optimization is critical to achieving the best possible parameters for a given system while keeping the cost minimal (Izci and Ekinci 2021). As optimization problems arise in many fields, such as engineering, science, economics, and business, there has been an enormous surge in the development of optimization algorithms (Gupta and Deep 2020). However, not all real-world optimization problems can be solved effectively using conventional mathematical programming approaches. Problems with unique characteristics such as non-differentiable nature, many decision variables, and objective functions, require alternative methods (Abualigah

✉ Laith Abualigah
aligah.2020@gmail.com

1 Department of Computer Engineering, Batman University, Batman 72100, Turkey

2 MEU Research Unit, Middle East University, Amman, Jordan

3 Vocational School of Social Sciences, Mus Alparslan University, Mus, Turkey

4 Center for Engineering Application and Technology Solutions, Ho Chi Minh City Open University, Ho Chi Minh City 700000, Vietnam

5 Soete Laboratory, Department of Electrical Energy, Metals, Mechanical Constructions, and Systems, Faculty of Engineering and Architecture, Ghent University, 9000 Ghent, Belgium

et al. 2021). Such challenges have led to the development new optimization techniques that can effectively solve complex problems that conventional methods cannot handle (Izci 2022). Therefore, metaheuristic algorithms have garnered immense attention as a powerful and effective alternative to tackle complex optimization problems (Gharehchopogh et al. 2023; Zaman and Gharehchopogh 2022; Gharehchopogh 2023; Shishavan and Gharehchopogh 2022; Mohammadzadeh and Gharehchopogh 2021a).

The realm of metaheuristic algorithms is characterized by an impressive diversity of approaches, each with unique strengths and limitations (Izci et al. 2023). These approaches, inspired by a range of physical and biological phenomena, are able to tackle complex optimization problems that cannot be effectively solved through conventional methods (Trojovský and Dehghani 2022). With their flexible and simple structure and the ability to avoid local optima through a random search, they have been widely studied and employed to solve a broad range of problems (Bharti et al. 2017, 2021; Ekinci et al. 2023a; Niu et al. 2022; Mohammadzadeh and Gharehchopogh 2021). Therefore, metaheuristic techniques have taken the scientific community by storm in the past few years, emerging as a prominent research area for tackling complex real-world problems. Their remarkable ability to provide optimal solutions in a computationally efficient manner and ease of implementation have made them a go-to tool for diverse engineering design applications (Carbas et al. 2021). Because the problem is considered a black box, and the algorithm attempts to solve it without being concerned about its nature (Izci et al. 2022). This property enables metaheuristic algorithms to be readily applicable to engineering optimization problems.

1.2 Motivation

The improved versions of HGS algorithm have been proposed in the literature and used to solve a variety of problems, indicating the feasibility of designing more efficient algorithms. However, achieving a balance between exploration and exploitation remains a challenge. Thus, there is a need to establish an effective balance strategy for optimization problems (Ekinci et al. 2023b). Also, it is important to note that each metaheuristic algorithm possesses unique strengths and weaknesses, and the "No Free Lunch theorem" (Wolpert and Macready 1997) asserts that there is no single algorithm that can optimally solve all problems (Agushaka et al. 2022; Ezugwu et al. 2022). Therefore, the motivation of this study is to develop an improved version of the HGS algorithm that is capable of effectively tackling a wide range of engineering optimization problems. In line with this motivation, this paper proposes an efficient solution: a novel HGS algorithm with elite opposite solution capability is proposed as an improved HGS algorithm. The improved version of the HGS algorithm proposed in this study uses pattern search (Torczon 1997) and elite opposition-based learning (Zhao et al. 2017) mechanisms. The latter is used to enhance the exploration, while the former is used to improve the exploitation of the HGS algorithm. To leverage superior performance, all those algorithms are embedded wisely within the HGS algorithm instead of running them sequentially.

1.3 Contributions

In line with the discussion provided in previous subsections, the performance of the improved HGS algorithm is initially evaluated across the CEC2019 test suite (Chakraborty et al. 2021) using moth-flame optimization (Mirjalili 2015a), capuchin search algorithm (Braik et al. 2021), Harris hawks optimization (Hussien et al. 2022), and arithmetic optimization algorithm (Zheng et al. 2021) for comparisons. Then the performance of the improved version of the HGS algorithm is tested on CEC2020 test suite (Snášel et al. 2023) as a more challenging platform. The latter platform is used to provide an insight on the contributions of each technique (elite OBL and PS) adopted in the improved HGS algorithm Besides, hybrid HGS algorithm with differential evolution, chaotic local search and evolutionary population dynamics techniques (Li et al. 2021) and quantum Nelder-Mead HGS algorithm (Xu et al. 2022) are also used as different reported variants of HGS algorithm. Furthermore, state-of-the-art algorithms of Runge–Kutta optimizer (Ahmadianfar et al. 2021), dwarf mongoose optimization (Agushaka et al. 2022), gazelle optimization algorithm (Agushaka et al. 2022b), and prairie dog optimization (Ezugwu et al. 2022) are also used to test the performance of the improved version of the HGS algorithm from a wider perspective. The related evaluations show the superior performance of the proposed method in terms of optimization capability. To further demonstrate the superiority of the improved version of the HGS algorithm proposed in this study, three different engineering design problems, with different natures and complexities, are also considered. In this regard, the identification of an IIR model (Mohammadi et al. 2022), training MLP (Agahian and Akan 2022), and designing a proportional-integral-derivative controller for a DFIG-based wind turbine system (Labdai et al. 2022) are considered in this paper in order to demonstrate the superior performance of the proposed method from a wider perspective of engineering design problems. To briefly list the contributions, the following remarks can be listed:

- A novel improved version of HGS algorithm is proposed by wisely integrating the elite OBL and PS mechanisms.

- Better performance of the proposed algorithm is demonstrated against CEC2019 and CEC2020 benchmark suites by using state-of the art algorithms.
- The proposed method is shown to have good performance for a wide range of real-world complex engineering problems know as identification of an IIR model, training MLP, and designing a proportional-integral-derivative controller for a DFIG-based wind turbine system.
- The superior performance for IIR identification model is shown comparatively using reported methods relying on cat swarm optimization (Panda et al. 2011), genetic algorithm (Panda et al. 2011), differential evolution algorithm (Yang et al. 2018), and opposition-based hybrid coral reefs optimization algorithm (Yang et al. 2018) alongside the original HGS algorithm.
- The performance of the proposed algorithm for training MLP is compared with the reported methods based on atom search optimization (Eker et al. 2021), dragonfly algorithm (Mirjalili 2016), bat optimization (Shehab et al. 2023), and grey wolf optimizer (Mirjalili 2015b) alongside the original HGS algorithm. The comparisons confirm the superior performance of the proposed method over the competitive methods in the literature.
- In terms of DFIG based wind turbine system, this work particularly utilizes the proposed algorithm in conjunction with the reptile search algorithm (Izci et al. 2022), gravitational search algorithm (Bharti et al. 2021), particle swarm optimization (Bharti et al. 2021), and bacterial foraging optimization (Bharti et al. 2021) which are the reported methods in the literature. The analyses show that the proposed method is also a good candidate for better controllability, further indicating the performance of the proposed method for a wide range of engineering applications.

2 Related works

2.1 Infinite impulse response system identification

Identifying infinite impulse response (IIR) models is critical in signal processing and system identification. IIR models are widely used to model systems with long-term memory, such as filters and control systems. They have a compact representation that can provide efficient computational solutions (Sharifi and Mojallali 2015; Singh et al. 2019). However, identifying IIR models can be challenging due to their nature, which requires specialized techniques such as maximum likelihood or subspace identification. Accurate identification of infinite impulse response models is essential for designing and optimizing systems that rely on them, such as digital signal processing and control systems (Zhao et al. 2020; Karaboga and Akay 2009). It can also lead to a better understanding and analysis of complex systems, enabling more effective and efficient decision-making.

Several metaheuristic algorithm examples for the IIR system identification problem can be found in the literature (Cuevas et al. 2023). For example, the application of genetic algorithm, particle swarm optimization, gravitational search algorithm, and inclined planes system optimization were explored for the design and optimization of digital IIR filters (Mohammadi et al. 2019). The performance and efficiency of these methods are evaluated using mean squared error, demonstrating the success of the research in achieving accurate results. In (Mohammadi et al. 2021), a novel approach was introduced for designing optimal IIR filters using a variable length particle swarm optimization algorithm with a weighted sum fitness function. The approach incorporates the filter order as a discrete variable in the particle vector to intelligently minimize the order and reduce the complexity of IIR filters. The proposed algorithm is evaluated through simulation results and demonstrates improved identified structures and performance. In another work, a metaheuristic algorithm called average differential evolution with local search was proposed for identifying optimal coefficients of unknown IIR systems (Durmuş 2022). By minimizing the error between the unknown system output and the adaptive IIR filter output, the proposed algorithm enables rapid convergence to global solutions in system identification problems, resulting in precise prediction of filter coefficients on multimodal error surfaces. The performance of average differential evolution with local search algorithm was demonstrated through comparisons with other methods, showing its efficiency in terms of convergence rate and mean square error value. It is feasible to extend the examples such as firefly algorithm (Upadhyay et al. 2016), teacher learner-based optimization algorithm (Singh et al. 2019), whale optimization algorithm (Luo et al. 2020), selfish herd optimization algorithm (Zhao et al. 2020) and bat algorithm (Kumar et al. 2016) as metaheuristic approaches reported for the IIR system identification problem.

2.2 Multilayer perceptron training

Training a multilayer perceptron (MLP) is a crucial task in machine learning and artificial neural networks. Multilayer perceptron is a feed-forward neural network consisting of multiple layers of interconnected nodes, with each layer performing a specific computation (Irmak et al. 2022). To train an MLP, the network weights and biases are adjusted based on the error between the predicted output and the actual output. This process, known as backpropagation, requires using an optimization algorithm, such as stochastic gradient

descent, to minimize the error and improve the model's accuracy. Training an MLP requires careful tuning of various hyperparameters, such as the learning rate and the number of hidden layers, to balance the trade-off between overfitting and underfitting. Successful training of an MLP can lead to highly accurate and reliable models that can be used for a wide range of applications, including image recognition, natural language processing, and speech recognition.

Several metaheuristics-based approaches have also been reported for the MLP training. For example, in a study, the authors proposed a hybrid training technique that combines the ant lion optimizer with MLP (Heidari et al. 2020). In the related study, an encoding scheme and objective formula were introduced, and the model was validated on sixteen standard datasets. Comparative experiments demonstrated that the proposed approach outperforms other well-known metaheuristic algorithms in terms of classification accuracy and convergence rates. In another study (Hong et al. 2020), a methodological approach for generating a landslide susceptibility map was presented by classifying landslide variables, weighting them using the certainty factor method, and optimizing the neural network's structural parameters using a genetic algorithm. The proposed model outperformed logistic regression and random forest models in terms of prediction accuracy, area under the curve, and relative landslide density, demonstrating its effectiveness as a spatial investigation tool for landslide susceptibility mapping. In Lee and Lee (2022), the researchers focused on predicting runoff in urban streams using an MLP with a harmony search optimizer. The proposed approach outperformed MLPs using other existing optimizers, resulting in more accurate runoff predictions with the smallest error compared to observed runoff peak values. The findings demonstrated the effectiveness of the proposed approach in accurately predicting urban stream runoff based on pump station discharge and rainfall information. In Li et al. (2022), improving the accuracy of medical data classification was investigated by using a modified biogeography-based optimization algorithm. The proposed algorithm incorporates different probability distributions into the migration process of biogeography-based optimization to enhance the performance and overcome issues such as local minimum, slow convergence, and sensitivity to initial values. Experimental results demonstrated that the proposed algorithm outperforms both the standard biogeography-based optimization algorithm and other adopted algorithms, leading to improved classification accuracy in medical data analysis. In a different study (Turkoglu and Kaya 2020), the application of the artificial algae algorithm was introduced for training artificial neural networks in various problem domains, particularly in classification tasks. Ten different classification datasets were used to test the performance of the proposed approach. The results indicated that the artificial algae algorithm is a reliable and effective approach for training artificial neural networks, demonstrating its potential as an alternative method for optimizing neural network parameters. In Lee (2023), the researchers focused on predicting the inflow of a centralized reservoir in an urban drainage system as a non-structural measure for preemptive operation. A new MLP model was proposed, which combined existing optimizers with an improved harmony search algorithm, resulting in improved accuracy compared to other existing optimizers in terms of mean square error and mean absolute error. The study in Bacanin et al. (2022) proposed an enhanced brainstorm optimization-based algorithm for the training. The algorithm demonstrated improved performance in terms of classification accuracy and convergence speed compared to other state-of-the-art approaches on a variety of benchmark datasets, outperforming them by 1–2% on average in terms of accuracy and dominating in terms of mean accuracy on the majority of datasets. In Bansal et al. (2019) a combination of an MLP with the lion optimization algorithm was proposed to optimize the architecture and training of the MLP for classification tasks. The proposed approach outperformed existing state-of-the-art techniques in terms of accuracy on various classification problems, showcasing its effectiveness in achieving improved performance. The study in Ma (2022) introduces an improved version of the moth-flame optimizer called adaptive moth flame optimization with opposition-based learning, which addresses the issues of slow convergence and local stagnation. The performance of the proposed algorithm was evaluated through benchmark function tests and compared to other algorithms in MLP training, demonstrating its superior accuracy, convergence rate, and classification performance. Lastly in Moghanian et al. (2020), an intrusion detection system that utilizes an artificial neural network trained using the grasshopper optimization algorithm was presented to improve accuracy in detecting network intrusion patterns. The proposed method demonstrated higher accuracy compared to state-of-the-art techniques in detecting abnormal and malicious traffic and attacks based on evaluation with different datasets.

2.3 Doubly fed induction generator-based wind turbine system control

The last engineering design problem considered for this paper is about controlling a doubly fed induction generator (DFIG) based wind turbine system, as it is crucial for ensuring the safe and efficient operation of the system. DFIG-based wind turbines are widely used due to their high efficiency and ability to generate power at variable speeds. Still, they are also highly complex systems that require sophisticated control algorithms to maintain stable operation (Labdai et al. 2022). The control system for a DFIG-based wind turbine must regulate the power output, voltage, and

frequency of the system while also protecting it from grid disturbances and other operational issues (Sudarsana Reddy and Mahalakshmi 2022). Failure to control the system effectively can result in voltage and frequency fluctuations, power losses, and even system failure, leading to downtime and reduced energy production. Effective control of doubly fed induction generator-based wind turbine systems is therefore essential for ensuring wind energy generation's reliability, safety, and economic viability.

Similar to the first two engineering design problems, metaheuristic approaches have also been employed for efficient control of DFIG-based wind turbine systems, as well. For example, in a study (Qouarti et al. 2023), the researchers focused on wind energy generation using DFIG, aiming to maximize power extraction from wind. Two control approaches were proposed: a combined control based on maximum power point tracking and sliding mode control, and a super twisting control based on particle swarm optimization and grey wolf optimization. Results demonstrated that the super twisting control tuned by particle swarm optimization algorithm outperforms other strategies in optimizing power extraction, as evidenced by comparing generator speed signals across different control scenarios. In another study (Mostafa et al. 2023), the challenge of operating a wind power system at the optimum power point, especially in the presence of uncertain wind speeds was addressed by using a metaheuristic optimization approach called driving training algorithm. Three maximum power point tracking scenarios were considered, and the proposed approach was shown to achieve efficient maximum power point tracking under different wind speeds compared to water cycle algorithm and particle swarm optimizer. In Benamor et al. (2019), a control strategy called root tree optimization was introduced to address chattering phenomena, minimize harmonic currents, and improve the performance of a DFIG system. The root tree optimization was utilized to adjust the parameters of a proportional-integral controller, resulting in improved dynamic and steady performance. The study in Palanimuthu et al. (2022) focuses on the design of fuzzy integral sliding mode control for a DFIG-based wind energy system using a membership function-dependent approach. The proposed approach utilized Takagi–Sugeno fuzzy modeling to represent the nonlinear DFIG-based wind energy system as a sum of local sub-models. A suitable FISMC was designed with a reaching law condition to handle disturbances, and a fuzzy-based Lyapunov function was constructed to evaluate the system's performance. The results showed that the membership function-dependent approach improves the performance index by 10% compared to the conventional approach, and simulation results demonstrated the stability and effectiveness of the proposed approach. In another work (Izci et al. 2022), reptile search algorithm was reported to tune a proportional-integral-derivative controller for improving the DFIG based wind turbine system's transient performance. Comparisons with other design approaches, such as gravitational search algorithm, bacterial foraging optimization, and particle swarm optimization, using the same controller, confirmed the enhanced efficiency and reliability of the system. In Muisyo et al. (2022), the application of a static synchronous compensator to enhance the low voltage ride-through capability of a 9 MW DFIG-based wind power plant during grid faults was investigated. The static synchronous compensator was tuned using the water cycle algorithm, particle swarm optimization, and a hybrid version of those algorithms. Simulation results showed that incorporating the static synchronous compensator with hybrid algorithm tuning effectively improves the wind power plant's low voltage ride-through capability, reducing voltage fluctuations and achieving better performance. In Ahmed et al. (2019), the researchers proposed the use of the Harris hawks algorithm for optimizing the tuning of integral classical controllers in load frequency control applications. The study focused on a two-area interconnected power system with a DFIG-based wind turbine in one area. By applying the Harris hawks algorithm, the performance of the system was enhanced, resulting in improved frequency and tie-line power oscillation damping, particularly when the DFIG participation level is high. With the work reported in Boureguig et al. (2023), artificial bee colony and grey wolf optimizer, were used for optimal feedback linearization control in a DFIG system. Simulation results using a 1.5 MW DFIG wind turbine demonstrated reduced overshoot, settling time, and steady-state error, indicating the effectiveness of the proposed approach. Moreover, a teaching learning-based optimization algorithm-assisted fractional order controller for the control of the rotor side converter and grid side converter in a DFIG-based wind turbine system was introduced (Karad and Thakur 2022). Simulation results demonstrated that the proposed approach outperforms the genetic algorithm and particle swarm optimization-based approaches in terms of DC link voltage control, solution time, and time domain performance parameters. Lastly, a swarm moth-flame optimizer was introduced in another work for optimizing the parameters of four interacting proportional-integral loops in a DFIG-based wind turbine (Huang et al. 2019). The proposed algorithm aimed to achieve maximum power point tracking and improved fault ride-through capability. The results of three case studies demonstrated that proposed approach outperforms existing metaheuristic techniques in terms of global convergence, optimal power tracking, and fault ride-through capability.

2.4 Improvement strategies for hunger games search algorithm

The hunger games search (HGS) algorithm (Yang et al. 2021) is one of the metaheuristic approaches that has been developed specifically to tackle specific optimization problems. Despite widespread use of HGS algorithm, critical issues, such as local minimal stagnation and immature convergence, remain to be addressed. Achieving a balance between exploration and exploitation stages is critical to enhancing the capabilities as the latter problems are caused by the randomized exploration and exploitation operators of metaheuristics, which necessitate the need for innovative strategies to mitigate or improve them (Luo et al. 2020; Gülcü 2022). In this regard, different enhancement strategies have been offered to improve the performance of the HGS algorithm.

For example, in (Kutlu Onay and Aydemir 2022), ten different chaotic maps applied to the HGS algorithm. The proposed chaotic HGS algorithm was evaluated on CEC2017 and 23 classical benchmark problems, as well as real engineering problems, such as cantilever beam design, tension/compression, and speed reducer. The results showed that chaotic HGS outperformed classical HGS and other state-of-the-art algorithms in the literature, indicating its promising potential for optimization tasks. In Nguyen and Bui (2021), the researchers proposed a novel soft computing model using HGS and artificial neural network and utilized it for predicting ground vibration intensity induced by mine blasting in the mining industry. They compared the performance of the proposed model with three other benchmark models based on different metaheuristic algorithms and found that the proposed model achieved the best results in terms of statistical criteria, thus, can be widely applied in open-pit mines to optimize blast patterns and minimize environmental effects. In Ma et al. (2022), the researchers developed a multi-strategy HGS and its binary variant by integrating HGS with a multi-strategy framework. The proposed algorithm was applied to global optimization and its binary variant to the feature selection problem. The experimental results showed that the proposed algorithm outperforms existing techniques in terms of classification accuracy, number of selected features, fitness values, and execution time, suggesting it to be a superior optimizer and a valuable feature selection technique. In Chakraborty et al. (2022), the researchers proposed a combination of the HGS algorithm with the whale optimization algorithm. By incorporating the hunger concept from HGS algorithm and the food searching techniques of whales, the proposed hybrid algorithm aims to address the limitations such as local optima trapping and premature convergence. Statistical analyses, complexity analysis, convergence analysis, and solving real-world engineering problems were conducted to demonstrate the efficacy of the newly designed algorithm. The results confirmed the improved performance of the proposed algorithm compared to other algorithms, supporting its effectiveness in optimization tasks. In Izci and Ekinci (2022), the researchers developed a novel control method for buck converter system. The proposed method involved the utilization of a fractional-order proportional-integral-derivative controller and the development of an improved version of the HGS algorithm. The proposed algorithm in the latter work was enhanced by incorporating the Nelder-Mead simplex method and a random learning mechanism in order to improve intensification and diversification abilities. The results confirmed the superiority of the proposed method for controlling a buck converter system in terms of performance, robustness, and effectiveness compared to the state-of-the-art approaches. In Abushanab et al. (2021), the researchers developed an artificial intelligence-based predictive model for friction stir welding of dissimilar polymeric materials. The related model combines a random vector functional link model with the HGS algorithm. Comparative analysis demonstrated that the proposed HGS algorithm-based model outperforms other models optimized with the state-of-the-art optimizers. In Izci et al. (2022c), the researchers developed a novel algorithm by incorporating a logarithmic spiral opposition-based learning technique into the HGS algorithm. The performance of the proposed algorithm was evaluated for both function optimization using benchmark functions from the CEC 2017 test suite and controller design for a magnetic ball suspension system. Comparative assessments were conducted, and the results demonstrated that the proposed algorithm outperforms other state-of-the-art methods, providing significant improvements in terms of transient response-related parameters and bandwidth for the magnetic ball suspension system. In Izci and Ekinci (2023), the researchers addressed the challenge of designing an effective controlling scheme for power converters. They proposed a novel approach by utilizing a fractional-order proportional-integral-derivative controller and a hybrid metaheuristic algorithm that combines HGS algorithm with simulated annealing. The proposed algorithm effectively tuned the controller, resulting in improved performance in terms of time and frequency domains, as well as disturbance rejection. Comparative analysis with other existing approaches further confirmed the superior performance of the proposed hybrid algorithm-based controlling scheme for a buck converter system. In Izci et al. (2022d), the researchers developed a novel algorithm by combining a modified opposition-based learning technique with the HGS algorithm. The developed algorithm was designed to tune a fractional order proportional-integral-derivative

controller for a magnetic ball suspension system. The algorithm's performance was evaluated using challenging benchmark functions from the CEC 2017 test suite, and its effectiveness in controlling the magnetic ball suspension system was demonstrated through various evaluations including statistical analysis, convergence profile, transient response, frequency response, disturbance rejection, and robustness. The results confirmed the superior ability of the proposed approach for controlling the magnetic ball suspension system.

3 HGS and proposed Imp-HGS algorithms

3.1 Hunger games search algorithm

Nature is filled with examples of animals surviving through hunger-driven activities. Their motions and behavioral choices are crucial to their survival. The Hunger Games Search (HGS) algorithm captures this reality and models it as a set of game rules, making it a powerful metaheuristic optimization tool (Yang et al. 2021). The game rules simulate cooperation between animals during foraging, accounting for their reluctance to cooperate. The HGS algorithm is represented mathematically by the game rule in Eq. (1):

$$\overline{X(t+1)} = \begin{cases} Game_1 : \overline{X(t)} \cdot (1 + randn(1)), & r_1 < C \\ Game_2 : \overline{W_1} \cdot \overline{X_b} + \overline{R} \cdot \overline{W_2} \cdot \left| \overline{X_b} - \overline{X(t)} \right|, & r_1 > C, r_2 > E \\ Game_3 : \overline{W_1} \cdot \overline{X_b} - \overline{R} \cdot \overline{W_2} \cdot \left| \overline{X_b} - \overline{X(t)} \right|, & r_1 > C, r_2 < E \end{cases} \tag{1}$$

Here, $randn(1)$ is a random number with normal distribution whereas r_1 and r_2 are two other random numbers within $[0, 1]$. t denotes the current iteration, $\overline{X_b}$ stands for the location of the best individual, $\overline{W_1}$ and $\overline{W_2}$ are the hunger weights and $\overline{X(t)}$ is the current iteration-based location of individuals.

The process of selecting between designated rules ($Game_1$, $Game_2$ and $Game_3$) is determined by the parameter C, while the $\overline{R}$ parameter is calculated as $\overline{R} = 2 \times shr \times rand - shr$ where $shr = 2 \times (1 - (t/T))$. The E parameter controls the variations of all positions and is defined as $E = sech(|F(i) - BFit|)$ where $F(i)$ is the fitness value of each individual, $BFit$ is the best fitness obtained so far, and $i \in 1, 2, \ldots, n$. The hyperbolic function, $sech(x) = (2/(e^x + e^{-x}))$, is used to calculate E.

The HGS algorithm consists of two search categories that simulate self-dependent individuals and teamwork to enhance diversification. Additionally, the algorithm mimics the starvation characteristics of each individual with the following equations, which are reflected in the hunger weights given by Eq. (1).

$$\overline{W_1(i)} = \begin{cases} hng(i) \cdot \frac{N}{Shng} \times r_4, & r_3 < C \\ 1, & r_3 > C \end{cases} \tag{2}$$

$$\overline{W_2(i)} = (1 - e^{-|hng(i) - Shng|}) \times r_5 \times 2 \tag{3}$$

These weights are expressed as a sum of the hungry feelings of all individuals denoted by $Shng$, while N represents the number of individuals. Equations (2) and (3) utilize r_3, r_4 and r_5, which are distinct random numbers selected from the range $[0, 1]$. The value of $hng(t)$ is determined by the following definition where $Allfit(i)$ represents the fitness of each individual in the current iteration.

$$hng(i) = \begin{cases} 0, & Allfit(i) == BFit \\ hng(i) + H, & Allfit(i) != BFit \end{cases} \tag{4}$$

The hunger sensation, H, is defined as follows, where $TH = ((F(i) - BFit)/(WF - BFit)) \times r_6 \times 2 \times (UB - LB)$.

$$H = \begin{cases} LH \times (1 + r), & TH < LH \\ TH, & TH \geq LH \end{cases} \tag{5}$$

In here, LH is the lower bound, and WF is the worst fitness value obtained so far. The feature space-related upper and lower bounds are respectively represented by UB and LB. Overall, the HGS algorithm strives to optimize the selection of rules and parameters to achieve desirable outcomes. Figure 1 provides a flowchart representing the logic of the HGS algorithm during the optimization tasks.

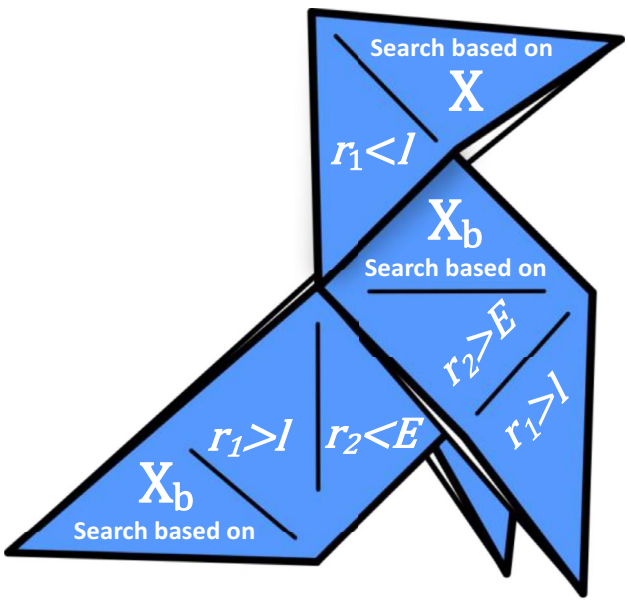


Fig. 1 Logic of HGS during optimization (Yang et al. 2021)

3.2 Improved hunger games search algorithm

3.2.1 Elite opposition-based learning

The original form of the opposition-based learning (OBL) technique (Tizhoosh 2005) has been a staple among researchers looking to enhance optimization algorithms (Izci et al. 2022c). The elite OBL is a unique approach within the realm of OBL that considers the best and current agents to generate opposite solutions of those agents (Izci et al. 2023b; Özmen et al. 2023). For the definition of elite OBL, $X^o = \langle x_1^o, x_2^o \dots, x_m^o \rangle$ can be used by considering $X = \langle x_1, x_2 \dots, x_m \rangle$ to be an elite candidate solution with m decision variables where $x_i^o = \delta(da_i + db_i) - x_i$, δ is a parameter within (0, 1), da_i and db_i are the dynamic boundaries that are defined as $da_i = min(x_i)$ and $db_i = max(x_i)$, respectively. In this study, three random variables of a, b and c, all of which are within [0, 1], are adopted to redefine the EOBL as $x_i^o = \delta(a \cdot da_i + b \cdot db_i) - c \cdot x_i$. The solution in elite OBL is kept within boundaries defined by $x_i^o = rand(Lb_i, Ub_i)$ where Lb_i is the lower and Ub_i is the upper limit, whereas $rand(Lb_i, Ub_i)$ is a random number within (Lb_i, Ub_i). Figure 2a provides a visual representation of the OBL mechanism.

3.2.2 Pattern search algorithm

The Pattern Search (PS) algorithm is an optimization technique that can find the optimal solution without using derivatives. The algorithm generates a point that may or may not be close to the solution. Around this point, the algorithm creates a collection of points called a mesh, updated as the algorithm progresses. The user defines the starting point for the search, and in the first iteration, the mesh size is considered as 1. The algorithm then constructs pattern vectors ($X_0 + [01]$, $X_0 + [10]$, $X_0 + [-10]$ and $X_0 + [0 - 1]$) and uses them to produce new mesh points. The objective function of each new point is calculated and compared to the current best point. If a better point is found, the search point is relocated to the new point, and the mesh size is expanded. If no better point is found, the mesh size is reduced, and the algorithm continues to search. This process continues until the optimal solution or termination condition is met. Figure 2 (b) demonstrates the mesh points and the search directions used in the PS mechanism.

3.2.3 Proposed algorithm

The improved hunger games search (Imp-HGS) algorithm proposed in this paper employs the elite opposition-based learning (OBL) mechanism (Zhao et al. 2017) to increase

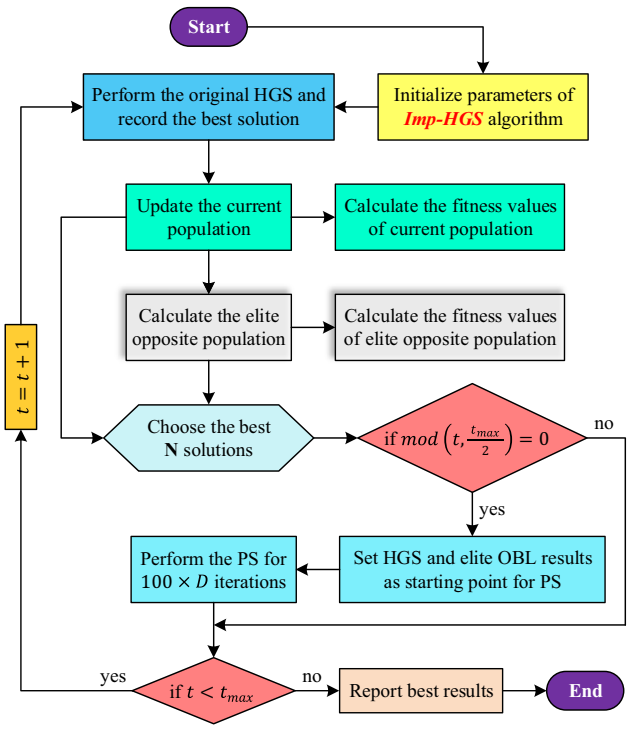
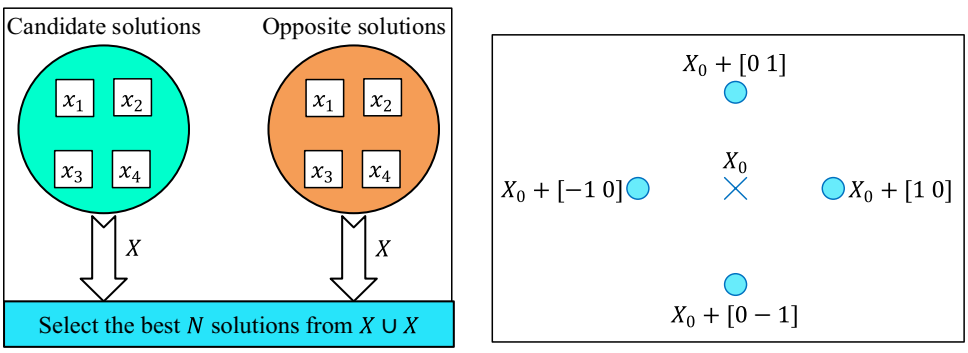


Fig. 3 Flowchart of proposed Imp-HGS algorithm

Fig. 2 OBL and PS mechanisms



the exploration performance of the original HGS algorithm. In contrast, it uses the pattern search (PS) mechanism (Torczon 1997) to enhance its exploitation capability further. Figure 3 demonstrates the process of the proposed Imp-HGS algorithm.

As observed from the detailed flowchart in the latter figure, the original HGS algorithm is aided by the modified elite OBL mechanism and the PS strategy. The Imp-HGS reaches further explorative performance with the modified elite OBL mechanism and exploitative performance with the PS mechanism. The Imp-HGS begins performing with the original HGS algorithm, and the produced best solution is further processed with the elite OBL mechanism. At this point, N best solutions are obtained, which are then used by the PS mechanism for reaching better exploitation. It is worth noting that the PS mechanism is not performed in each iteration; instead, it operates two times during the entire process and runs for $100 \times D$ iterations where D is the dimension size of the problem. Such a processing style has been decided after extensive simulations. Consequently, the ability of the original HGS algorithm is improved significantly with the design proposed in this work. In the proposed method, all adopted algorithms are used with their default parameter values in order to have a have a fair conclusion of the performance during the optimization tasks. It is also worth noting that the variable parameters, population size and the number of iterations have certain limitations on the performance of the proposed algorithm for this specific problem. Choosing bigger population sizes and iteration numbers will have a considerable increase on the performance of the proposed method at the expense of relatively higher computational load, however, after a certain limit there will be no increase.

4 Experimental results on benchmark functions

4.1 CEC2019 benchmark functions

To evaluate the performance of the Imp-HGS algorithm, challenging test functions from the CEC2019 test suite were adopted. The details of those test functions are provided in Table 1. As the CEC2019 test suite is composed of the complex and highly difficult benchmark set, it can be used as a good test bed for the overall performance evaluation of the algorithms. This set of challenging and complex functions is specifically designed to push the limits of optimization algorithms and test their robustness and performance under extreme conditions. With its innovative and cutting-edge approach, CEC2019 sets a new standard for benchmarking in the optimization community and provides a crucial tool for advancing the state-of-the-art in this field.

For the statistical comparison purpose, moth-flame optimization (MFO) (Mirjalili 2015a), capuchin search algorithm (CapSA) (Braik et al. 2021), Harris hawks optimization (HHO) (Heidari et al. 2019) and arithmetic optimization algorithm (AOA) (Abualigah et al. 2021) are used in this study. The default parameter values for the latter listed algorithms are adopted during the tests. For the optimization of the CEC2019 test suite, a total iteration of 500 and a population size of 30 was used, and each of the algorithms was run 30 times in order to provide a fair comparison. Table 2 lists the corresponding values of the statistical metrics of average (mean), standard deviation, and best and worst; the bold font refers to the best results in the tables. From the observation of this table, one can see that the proposed Imp-HGS algorithm demonstrated consistently superior performance compared to the other algorithms used for comparison across the CEC 2019 test functions (F1–F10). With the lowest average values for all functions, Imp-HGS exhibited a remarkable level of efficiency in optimizing the objective functions.

Table 1 Details of CEC2019 benchmark function problems

Function	Definition	Dimension	Search range	F_{min}
F1	Store's Chebyshev Polynomial fitting problem	9	[−8192, 8192]	1
F2	Inverse Hilbert matrix problem	16	[−16384, 16384]	1
F3	Lennard–Jones minimum energy cluster	18	[−4, 4]	1
F4	Rastrigin's function	10	[−100, 100]	1
F5	Griewangk's function	10	[−100, 100]	1
F6	Weierstrass function	10	[−100, 100]	1
F7	Modified Schwefel's function	10	[−100, 100]	1
F8	Expanded Schaffer's F6 function	10	[−100, 100]	1
F9	Happy cat function	10	[−100, 100]	1
F10	Ackley function	10	[−100, 100]	1

Table 2 Comparative statistical results of Imp-HGS, HGS, MFO, CapSA, HHO and AOA algorithms on CEC2019 benchmark functions

Function	Measure	Imp-HGS	HGS	MFO	CapSA	HHO	AOA
F1	Average	**3.7522E+04**	4.6396E+04	1.5905E+10	4.0515E+04	5.3493E+04	1.1947E+10
	Standard deviation	2.7306E+02	1.2413E+04	2.5790E+10	2.4563E+03	5.0620E+03	4.7159E+10
	Best	3.7274E+04	3.8933E+04	3.3887E+08	3.7472E+04	4.3994E+04	1.0687E+06
	Worst	3.8837E+04	1.0598E+05	1.0233E+11	4.6397E+04	6.7651E+04	2.5541E+11
F2	Average	**1.7343E+01**	**1.7343E+01**	**1.7343E+01**	**1.7343E+01**	1.7363E+01	1.9338E+01
	Standard deviation	0.0000E+00	3.0403E−12	0.0000E+00	7.2183E−05	7.6387E−03	4.0087E−01
	Best	1.7343E+01	1.7343E+01	1.7343E+01	1.7343E+01	1.7350E+01	1.8277E+01
	Worst	1.7343E+01	1.7343E+01	1.7343E+01	1.7343E+01	1.7381E+01	1.9848E+01
F3	Average	**1.2702E+01**	**1.2702E+01**	**1.2702E+01**	**1.2702E+01**	1.2702E+01	1.2703E+01
	Standard deviation	0.0000E+00	9.2814E−08	2.0726E−05	9.0336E−15	6.3218E−06	1.1612E−03
	Best	1.2702E+01	1.2702E+01	1.2702E+01	1.2702E+01	1.2702E+01	1.2702E+01
	Worst	1.2702E+01	1.2702E+01	1.2703E+01	1.2702E+01	1.2702E+01	1.2707E+01
F4	Average	**4.7559E+01**	5.0826E+01	1.5089E+02	5.9068E+01	2.0363E+02	1.2631E+04
	Standard deviation	2.1375E+01	3.1281E+01	3.2124E+02	2.4608E+01	8.8490E+01	5.4663E+03
	Best	1.1940E+01	1.9899E+01	8.9546E+00	1.5925E+01	1.0431E+02	5.1739E+03
	Worst	9.5513E+01	1.5135E+02	1.6394E+03	1.2039E+02	4.5536E+02	2.7708E+04
F5	Average	**1.1591E+00**	1.2059E+00	1.2124E+00	1.2767E+00	2.3072E+00	4.2574E+00
	Standard deviation	9.2261E−02	1.2560E−01	1.6758E−01	1.5150E−01	5.7249E−01	7.6269E−01
	Best	1.0099E+00	1.0591E+00	1.0246E+00	1.0615E+00	1.4616E+00	2.7547E+00
	Worst	1.3815E+00	1.5236E+00	1.7031E+00	1.5513E+00	4.1840E+00	5.9830E+00
F6	Average	**4.2429E+00**	4.5001E+00	5.7881E+00	8.0787E+00	9.7227E+00	8.9082E+00
	Standard deviation	1.6152E+00	9.7008E−01	1.9662E+00	1.7223E+00	1.0071E+00	8.2634E−01
	Best	1.0978E+00	2.9792E+00	1.2029E+00	3.8379E+00	7.3419E+00	7.0610E+00
	Worst	6.9713E+00	6.7850E+00	1.0510E+01	1.1767E+01	1.1345E+01	1.0286E+01
F7	Average	**1.7742E+02**	1.9838E+02	3.6743E+02	4.3188E+02	4.2885E+02	2.2944E+02
	Standard deviation	1.0002E+02	1.4914E+02	2.3483E+02	3.4377E+02	2.2215E+02	1.2945E+02
	Best	5.6151E+00	−9.0789E+01	−4.4316E+01	−8.6995E+01	9.4918E+01	4.5871E+00
	Worst	3.3603E+02	4.6596E+02	9.2329E+02	1.1915E+03	1.0436E+03	5.9381E+02
F8	Average	**5.1140E+00**	5.3271E+00	5.5732E+00	5.2924E+00	5.8393E+00	5.5068E+00
	Standard deviation	6.2273E−01	5.7782E−01	6.4546E−01	9.6371E−01	4.9908E−01	5.2068E−01
	Best	3.4790E+00	3.7203E+00	3.7665E+00	2.9771E+00	4.5395E+00	3.7938E+00
	Worst	5.8863E+00	6.0650E+00	6.6660E+00	6.5448E+00	6.5552E+00	6.1256E+00
F9	Average	**2.6417E+00**	3.2286E+00	2.8133E+00	2.7550E+00	3.5288E+00	8.9314E+02
	Standard deviation	2.2480E−01	4.4220E−01	2.6132E−01	3.6140E−01	6.2553E−01	4.7130E+02
	Best	2.3985E+00	2.5385E+00	2.4580E+00	2.4305E+01	2.6864E+00	1.8149E+02
	Worst	2.9496E+00	4.7182E+00	3.5594E+00	4.2005E+01	5.1537E+00	2.1622E+03
F10	Average	**1.9998E+01**	2.0014E+01	2.0208E+01	2.0122E+01	2.0236E+01	2.0131E+01
	Standard deviation	8.6117E−03	2.0265E−02	1.5969E−01	9.8301E−02	1.1977E−01	7.3199E−02
	Best	1.9955E+01	1.9999E+01	1.9999E+01	2.0009E+01	2.0033E+01	1.9867E+01
	Worst	2.0008E+01	2.0064E+01	2.0542E+01	2.0393E+01	2.0459E+01	2.0280E+01

This indicates that Imp-HGS consistently converged towards optimal solutions and achieved highly competitive results. In contrast, the other algorithms (HGS, MFO, CapSA, HHO, and AOA) displayed higher average values, suggesting a comparatively less efficient performance. The consistent dominance of Imp-HGS across all functions underscores its effectiveness and highlights it as a promising choice for solving optimization problems.

In addition to achieving the lowest average values, the Imp-HGS algorithm also demonstrated notable performance in other metrics when compared to the other algorithms across the CEC 2019 test functions (F1 to F10). Firstly, the Imp-HGS algorithm consistently exhibited lower standard deviation values, indicating a higher level of stability and robustness in its optimization process. This suggests that Imp-HGS consistently generated solutions that were closer

to each other, resulting in a more reliable and predictable optimization outcome.

Furthermore, the Imp-HGS algorithm consistently achieved competitive results in terms of the best and worst values. While it may not always achieve the absolute best value for a given function, it consistently performed well and approached the optimal solutions. On the other hand, the worst values obtained by Imp-HGS were consistently better than or comparable to the other algorithms, indicating that even in worst-case scenarios, Imp-HGS produced solutions that were relatively close to the optimal or acceptable range.

Taken together, the Imp-HGS algorithm's performance in terms of standard deviation, best, and worst values further reinforces its efficacy and reliability. Its ability to consistently provide low standard deviation values, competitive best results, and satisfactory worst results indicates its capability to explore and exploit the search space effectively, leading to a robust and efficient optimization process. In addition, the convergence curves provided in Fig. 4 further supports the claim that the proposed Imp-HGS algorithm is efficient and can reach better solutions.

4.2 CEC2020 benchmark functions

As part of the effort to demonstrate the more excellent performance of the proposed Imp-HGS algorithm, benchmark functions from CEC2020 test suite have also been adopted for this study. This test suite is used as a challenging platform for performance evaluation of optimization algorithms. Table 3 lists the details of the CEC2020 benchmark functions (F1_CEC2020 to F10_CEC2020).

The performance of the Imp-HGS algorithm on CEC2020 benchmark functions was performed to provide an insight on the contributions of each technique (elite OBL and PS) adopted in Imp-HGS algorithm. Besides, hybrid HGS algorithm with differential evolution, chaotic local search and evolutionary population dynamics techniques (DECEHGS) (Li et al. 2021) and quantum Nelder-Mead HGS (IHGS) algorithm (Xu et al. 2022) are used in this study as different reported variants of HGS algorithm. Furthermore, state-of-the-art algorithms of Runge–Kutta optimizer (RUN) (Ahmadianfar et al. 2021), dwarf mongoose optimization (DMO) algorithm (Agushaka et al. 2022), gazelle optimization algorithm (GOA) (Agushaka et al. 2022b), and prairie dog optimization (PDO) algorithm (Ezugwu et al. 2022) are also used to test the performance of the proposed Imp-HGS algorithm from a wider perspective. The results listed in Table 4 are obtained with 30 independent runs using a population size of 50 and maximum number of iterations of 1000 for fair comparison.

As demonstrated in Table 4, the proposed Imp-HGS algorithm has the best statistical performance on these test functions when it is being used with the integration of both elite OBL and PS mechanisms. When considering the average performance, Imp-HGS consistently achieves competitive results. Across different test functions (F1–F10), the average values obtained by Imp-HGS are comparable to or better than other algorithms. This suggests that Imp-HGS demonstrates robustness and effectiveness in finding good solutions for a wide range of optimization problems.

Similarly, when examining the best and worst solutions found, the Imp-HGS algorithm consistently performs well. It is able to find high-quality solutions, as evidenced by the best solution values obtained, which are comparable to or better than other algorithms. Additionally, the worst solution values obtained by Imp-HGS are generally lower than those of other algorithms, indicating a more reliable and stable performance. In terms of standard deviation, which measures the spread or variability of solutions, the Imp-HGS algorithm demonstrates favorable results. The standard deviation values obtained by Imp-HGS are generally lower or comparable to other algorithms. A lower standard deviation indicates that the algorithm produces more consistent and reliable results.

Overall, the better performance of the Imp-HGS algorithm in the CEC 2020 test suite can be attributed to its ability to effectively explore the solution space, find high-quality solutions, and maintain stability and consistency in its optimization process. The algorithm's balance between exploration and exploitation enables it to efficiently navigate complex landscapes and converge to promising solutions.

5 Experimental results on IIR model identification

5.1 IIR filter design

System identification refers to the representation of an unknown system mathematically by considering input and output data. An optimization algorithm is used to minimize an error function (between the candidate model's output and the actual plant's output) in order to obtain an optimal model for the unknown plant. On the other hand, fewer model parameters can be used via infinite impulse response models to meet the performance specifications and produce a more accurate representation of physical plants for real-world applications (Mohammadi et al. 2022). An arbitrary system's infinite impulse response identification model is illustrated in Fig. 5, where $y(k)$ and $d(k)$ respectively represent the output of the infinite impulse response filter and the unknown

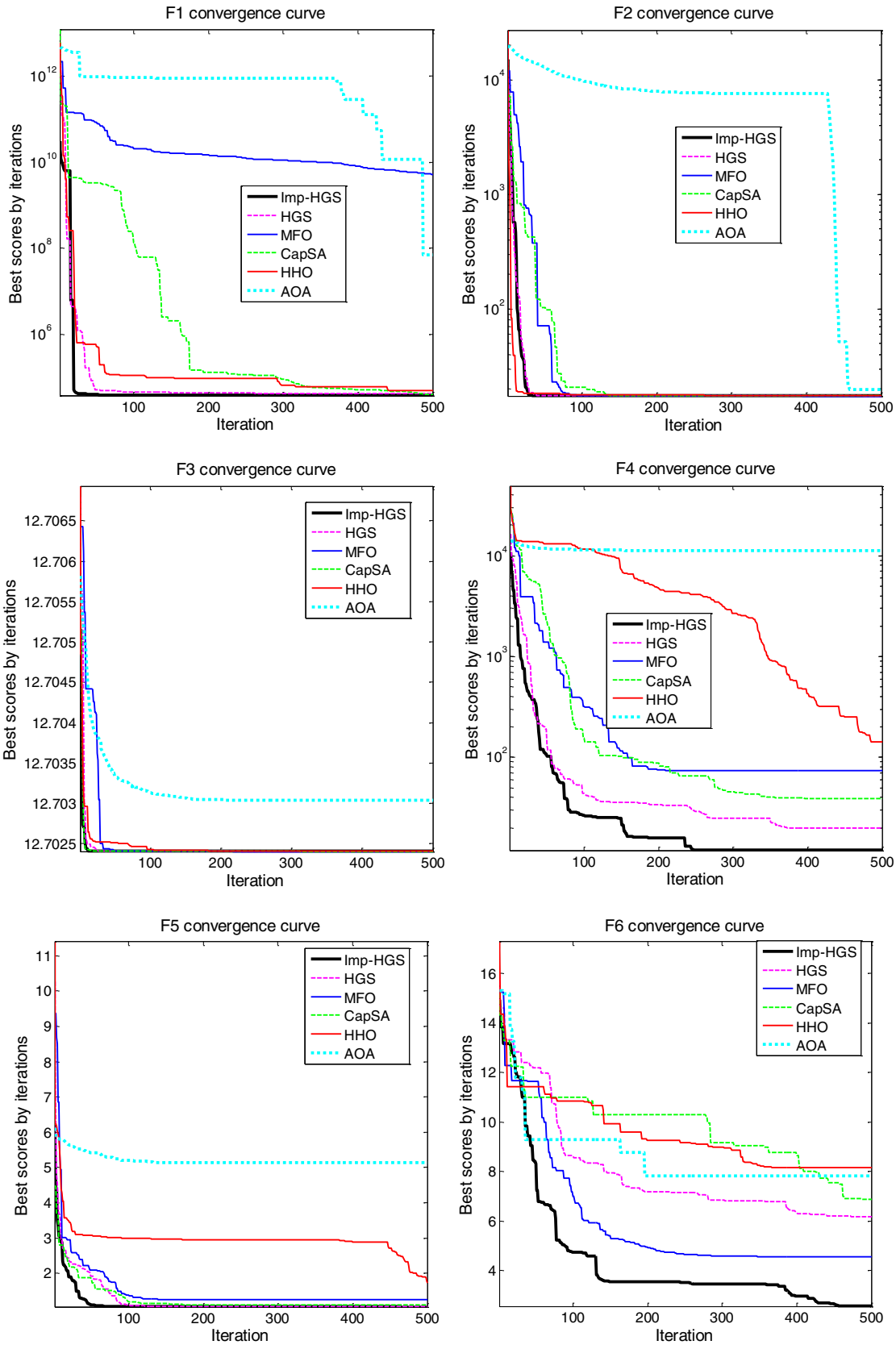


Fig. 4 Comparative convergence curves obtained for CEC2019 benchmark fuctions

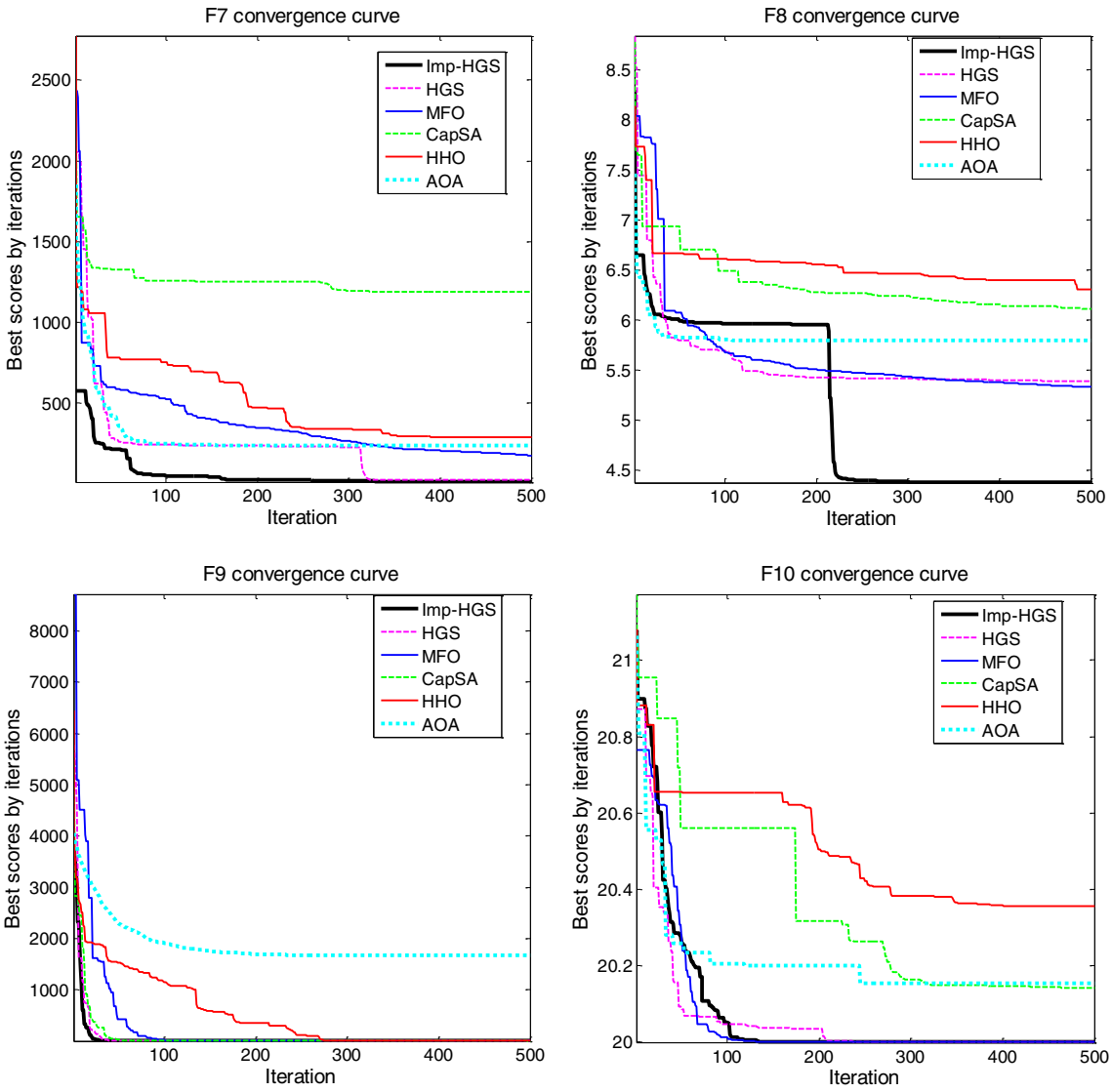


Fig. 4 (continued)

Table 3 Details of CEC2020 benchmark function problems

Function	Definition	Dimension	Search range	F_{min}
F1_CEC2020	Shifted and rotated bent cigar function	10	[−100, 100]	100
F2_CEC2020	Shifted and rotated Schwefel's function	10	[−100, 100]	1100
F3_CEC2020	Shifted and rotated Lunacek bi-Rastrigin function	10	[−100, 100]	700
F4_CEC2020	Expanded Rosenbrock's plus Griewank's function	10	[−100, 100]	1900
F5_CEC2020	Hybrid function 1 ($N=3$)	10	[−100, 100]	1700
F6_CEC2020	Hybrid function 2 ($N=4$)	10	[−100, 100]	1600
F7_CEC2020	Hybrid function 3 ($N=5$)	10	[−100, 100]	2100
F8_CEC2020	Composition function 1 ($N=3$)	10	[−100, 100]	2200
F9_CEC2020	Composition function 2 ($N=4$)	10	[−100, 100]	2400
F10_CEC2020	Composition function 3 ($N=5$)	10	[−100, 100]	2500

Table 4 Statistical results of CEC-2020 test functions

Function	Measure	Imp-HGS	OBL-HGS	HGS-PS	DECEHGS	IHGS	RUN	DMO	GOA	PDO
F1_CEC2020	Average	**100.33**	132.87	133.55	149.71	144.07	1190	100.43	100.43	100.46
	Standard deviation	0.51685	35.963	31.112	47.084	42.815	1595.8	0.5024	0.35345	0.35113
	Best	100	100.42	101.42	100.03	100	100	100.06	100.04	100.06
	Worst	101.49	208.15	173.03	231.98	203.61	3776.6	101.8	101.51	101.58
F2_CEC2020	Average	**1104.7**	1108.6	1123.1	1109.7	1119.5	1193.9	1126.3	1246.4	1241.4
	Standard deviation	7.4238	8.4742	31.783	8.3848	27.867	61.191	96.47	107.35	63.833
	Best	1100	1100	1100	1100	1100	1107.1	1101.4	1106.6	1130.8
	Worst	1116.8	1117.1	1218.4	1117.1	1218.4	1367.9	1347	1495.8	1375.1
F3_CEC2020	Average	**707.89**	709.99	711.19	712.71	711.26	713.85	733.22	719.59	719.69
	Standard deviation	2.113	4.3835	3.4128	3.9634	3.0441	3.27	5.1087	3.0154	2.963
	Best	701.99	700.99	706.01	707.85	706.35	706.94	715.3	711.23	714.03
	Worst	711.83	718.7	718.58	720.85	716.44	720.27	742.87	724.6	724.97
F4_CEC2020	Average	**1900**	**1900**	**1900**	**1900**	**1900**	**1900**	1901.2	1901.2	1901.2
	Standard deviation	0	0	0	0	0	0	0.54311	0.31242	0.3165
	Best	1900	1900	1900	1900	1900	1900	1900.3	1900.5	1900.6
	Worst	1900	1900	1900	1900	1900	1900	1902.5	1901.8	1901.8
F5_CEC2020	Average	1725.2	1737	1744.6	1728.2	1729.1	**1712.6**	1749	1756.1	1751.3
	Standard deviation	12.499	25.788	19.378	17.748	10.359	10.509	18.67	16.75	16.644
	Best	1700	1701	1724.7	1701.6	1705.6	1702	1715.3	1734.7	1720.1
	Worst	1742.7	1789	1798.7	1755.7	1750.8	1730.8	1819.2	1799.6	1797.9
F6_CEC2020	Average	**1600**	1601.7	1600.1	1604.1	1600.2	1601.4	**1600**	**1600**	1600.5
	Standard deviation	0	5.2003	0.21772	7.4208	0.25244	0.83913	0.000726	0.000893	0.001129
	Best	1600	1600	1600	1600	1600	1600.5	1600	1600	1600.5
	Worst	1600	1621	1600.7	1620.4	1600.8	1602.9	1600	1600	1600.5
F7_CEC2020	Average	**2101.9**	2127.7	2135.6	2123.7	2111.9	2343.2	2109.3	2110	2108
	Standard deviation	5.6087	50.9	55.146	48.15	36.114	211.95	9.74	5.5831	4.5969
	Best	2100	2100	2100	2100	2100	2131.4	2103.8	2102.9	2101.5
	Worst	2118.4	2218.4	2218.4	2218.4	2218.4	2743.6	2178.5	2122.5	2122.5
F8_CEC2020	Average	**2233.6**	2260.2	2259.3	2255.7	2256.7	2241.1	2269.4	2286.7	2278.3
	Standard deviation	42.158	45.954	47.705	47.618	43.884	44.057	14.228	35.252	43.878
	Best	2200	2200	2200	2200	2209.4	2200	2200.1	2200.2	2200
	Worst	2302.1	2304.2	2306.8	2305.6	2306.3	2303.9	2303.3	2303.5	2304.4
F9_CEC2020	Average	2557.9	2652.4	2634.6	2747.7	2748.2	2747.9	2525	**2523.3**	2559.5
	Standard deviation	21.029	112.03	100.51	67.854	76.021	8.9794	27.203	67.32	83.741
	Best	2500	2500	2546.6	2500	2500	2735.6	2500	2418.4	2500
	Worst	2575.8	2763.7	2759.3	2778.4	2808.6	2769.3	2590.2	2734.1	2739.7
F10_CEC2020	Average	**2768.3**	2818.3	2842.7	2844.2	2845.8	2848.8	2803.2	2871.8	2887.9
	Standard deviation	115.18	62.952	14.47	12.027	8.6598	7.6462	9.2352	80.727	53.863
	Best	2600	2600	2800	2800	2800	2847.4	2598.1	2601.3	2602.7
	Worst	2847.4	2847.4	2847.5	2847.5	2847.5	2889.3	2844.4	2897.8	2897.8

plant. On the other hand, $x(k)$ stands for the applied input signal whereas m and n are respectively the coefficients of the numerator and denominator that are described in the following subsection.

5.2 Experimental setup and IIR model identification results

The following form represents the transfer function of an infinite impulse response (IIR) system considering the details provided in the previous subsection.

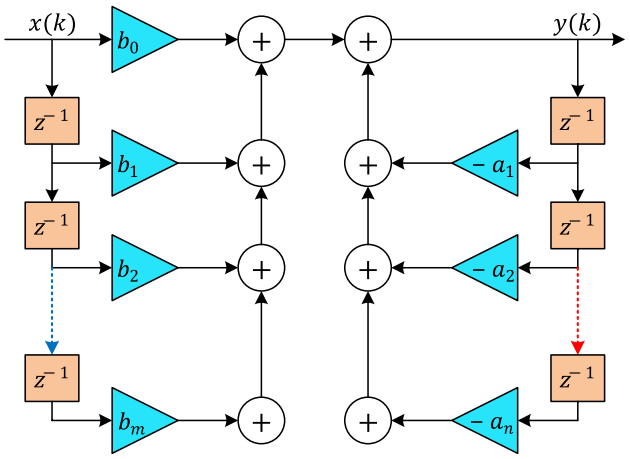


Fig. 5 Detailed structure of IIR filter

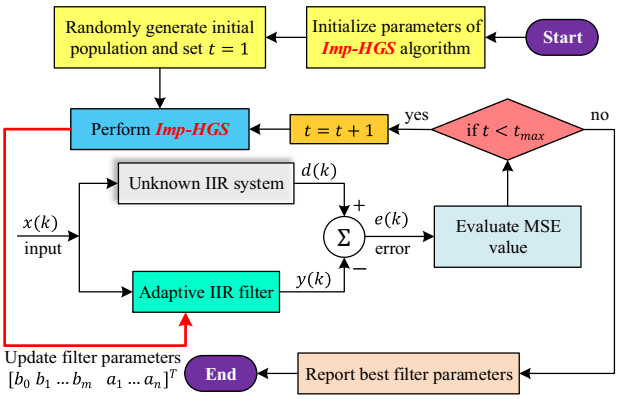


Fig. 6 Imp-HGS algorithm-based IIR system identification problem

$$\frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_m z^{-m}}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n}} \tag{6}$$

Here, the pole and zero parameters of the infinite impulse response model are denoted by a_i and b_j where $i = 1, 2, \ldots, n$ and $j = 0, 1, \ldots, m$. The difference equation form of the transfer function can be written as follows where x (k) and $y(k)$ represent the input and the output of the filter, respectively.

$$y(k) + \sum_{i=1}^{n} a_i \cdot y(k-i) = \sum_{j=0}^{m} b_j \cdot x(k-j) \tag{7}$$

Figure 6 demonstrates the block diagram of an adaptive IIR system identification system designed via the Imp-HGS algorithm. Here, $e(k)$ denotes the error between the model and the actual plant ($e(k) = d(k) - y(k)$), which can be used for representing the IIR model identification problem as a minimization problem using mean squared error (MSE) given in the following definition where W is the number of samples employed in the simulation.

$$f(\theta) = \frac{1}{W} \sum_{k=1}^{W} (d(k) - y(k))^2 \tag{8}$$

This work uses four benchmark examples of the IIR system identification presented in Table 5. In the experiments, the parameters of the algorithm are set as 30 runs, 1000 total iteration numbers (t_{max}) and 30 population size. The benchmark examples' statistical results are compared with the available literature. In this context, cat swarm optimization

Table 5 Four digital IIR system identification problems

Test system		Transfer function
Example I	Plant	$H_P(z^{-1}) = \frac{0.05-0.4z^{-1}}{1-1.1314z^{-1}+0.25z^{-2}}$
	Same order model	$H_S(z^{-1}) = \frac{b_0+b_1z^{-1}}{1-a_1z^{-1}-a_2z^{-2}}$
	Reduced order model	$H_R(z^{-1}) = \frac{b_0}{1-a_1z^{-1}}$
Example II	Plant	$H_P(z^{-1}) = \frac{-0.2-0.4z^{-1}+0.5z^{-2}}{1-0.6z^{-1}+0.25z^{-2}-0.2z^{-3}}$
	Same order model	$H_S(z^{-1}) = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{1-a_1z^{-1}-a_2z^{-2}-a_3z^{-3}}$
	Reduced order model	$H_R(z^{-1}) = \frac{b_0+b_1z^{-1}}{1-a_1z^{-1}-a_2z^{-2}}$
Example III	Plant	$H_P(z^{-1}) = \frac{1-0.9z^{-1}+0.81z^{-2}-0.729z^{-3}}{1+0.04z^{-1}+0.2775z^{-2}-0.2101z^{-3}+0.14z^{-4}}$
	Same order model	$H_S(z^{-1}) = \frac{b_0+b_1z^{-1}+b_2z^{-2}+b_3z^{-3}}{1-a_1z^{-1}-a_2z^{-2}-a_3z^{-3}-a_4z^{-4}}$
	Reduced order model	$H_R(z^{-1}) = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{1-a_1z^{-1}-a_2z^{-2}-a_3z^{-3}}$
Example IV	Plant	$H_P(z^{-1}) = \frac{0.1084+0.5419z^{-1}+1.0837z^{-2}+1.0837z^{-3}+0.5419z^{-4}+0.1084z^{-5}}{1+0.9853z^{-1}+0.9738z^{-2}+0.3864z^{-3}+0.1112z^{-4}+0.0113z^{-5}}$
	Same order model	$H_S(z^{-1}) = \frac{b_0+b_1z^{-1}+b_2z^{-2}+b_3z^{-3}+b_4z^{-4}+b_5z^{-5}}{1-a_1z^{-1}-a_2z^{-2}-a_3z^{-3}-a_4z^{-4}-a_5z^{-5}}$
	Reduced order model	$H_R(z^{-1}) = \frac{b_0+b_1z^{-1}+b_2z^{-2}+b_3z^{-3}+b_4z^{-4}}{1-a_1z^{-1}-a_2z^{-2}-a_3z^{-3}-a_4z^{-4}}$

Table 6 The statistical results obtained from the IIR system identification benchmark examples

Test system	Model	Measure	Imp-HGS	HGS	CSO	GA	DE	OHCRO
Example I	Same order	Average	**4.44E−34**	5.44E−05	6.38E−05	1.47E−03	6.36E−09	7.53E−11
		Standard deviation	8.23E−34	8.07E−05	2.89E−07	1.55E−03	2.30E−09	2.62E−10
		Best	0	4.21E−10	6.36E−05	2.64E−04	9.63E−10	1.93E−15
		Worst	2.14E−33	2.76E−04	6.46E−05	4.92E−03	9.78E−09	1.48E−09
	Reduced order	Average	**1.21E−02**	1.25E−02	1.75E−02	4.69E−02	1.64E−01	1.60E−01
		Standard deviation	2.29E−04	4.30E−04	4.91E−18	1.32E−02	1.31E−02	1.34E−02
		Best	1.14E−02	1.14E−02	1.75E−02	2.71E−02	1.40E−01	1.32E−01
		Worst	1.24E−02	1.37E−02	1.75E−02	5.78E−02	1.85E−01	1.95E−01
Example II	Same order	Average	**2.10E−34**	1.75E−04	6.35E−05	2.51E−03	1.15E−10	1.49E−17
		Standard deviation	2.89E−34	1.75E−04	1.69E−18	1.49E−03	2.31E−11	1.89E−17
		Best	0	1.18E−09	6.35E−05	7.32E−04	6.51E−11	5.91E−19
		Worst	7.19E−34	5.36E−04	6.35E−05	6.15E−03	1.48E−10	6.18E−17
	Reduced order	Average	**5.81E−04**	6.39E−04	1.39E−03	3.26E−02	2.65E−03	2.80E−03
		Standard deviation	2.62E−05	4.77E−05	1.08E−19	1.61E−02	3.87E−04	5.84E−04
		Best	5.08E−04	5.20E−04	1.39E−03	1.65E−02	1.84E−03	1.51E−03
		Worst	6.10E−04	7.26E−04	1.39E−03	6.67E−02	3.71E−03	4.25E−03
Example III	Same order	Average	**4.52E−13**	3.01E−03	5.94E−05	1.74E−02	7.91E−09	2.57E−09
		Standard deviation	2.09E−12	2.82E−03	8.30E−09	1.23E−02	1.53E−09	1.10E−08
		Best	2.06E−22	3.61E−05	5.94E−05	7.16E−03	3.64E−09	7.78E−13
		Worst	1.14E−11	8.92E−03	5.94E−05	4.49E−02	9.97E−09	7.27E−08
	Reduced order	Average	3.57E−03	8.46E−03	6.71E−03	4.66E−02	2.62E−02	**2.58E−03**
		Standard deviation	1.99E−04	2.67E−03	3.00E−11	2.33E−02	3.33E−03	2.61E−04
		Best	3.08E−03	4.22E−03	6.71E−03	1.94E−02	2.04E−02	1.86E−03
		Worst	3.83E−03	1.19E−02	6.71E−03	9.25E−02	3.48E−02	3.21E−03
Example IV	Same order	Average	**2.77E−06**	2.37E−04	6.39E−05	3.40E−02	NR	NR
		Standard deviation	2.68E−06	1.89E−04	2.90E−07	1.48E−02	NR	NR
		Best	2.55E−07	2.62E−06	6.36E−05	1.33E−02	NR	NR
		Worst	8.52E−06	6.03E−04	6.45E−05	6.42E−02	NR	NR
	Reduced order	Average	**3.65E−05**	3.76E−04	7.86E−05	3.24E+04	NR	NR
		Standard deviation	9.08E−06	2.30E−04	2.50E−05	9.68E+04	NR	NR
		Best	1.54E−05	5.60E−05	6.95E−05	8.46E−02	NR	NR
		Worst	5.21E−05	8.21E−04	1.45E−04	2.90E+05	NR	NR

NR not reported

(CSO) (Panda et al. 2011), genetic algorithm (GA) (Panda et al. 2011), differential evolution (DE) algorithm (Yang et al. 2018) and opposition-based hybrid coral reefs optimization (OHCRO) algorithm (Yang et al. 2018) are used in this study alongside the original HGS algorithm. The results presented are presented in Table 6.

The statistical results in Table 6 demonstrate the performance of various algorithms, including Imp-HGS, HGS, CSO, GA, DE, and OHCRO. In terms of the average values, Imp-HGS consistently outperformed HGS, CSO, GA, DE, and OHCRO across all the benchmark examples. This suggests that Imp-HGS was able to achieve lower average errors or losses in the system identification process compared to the other algorithms. Furthermore, when considering the standard deviation, Imp-HGS exhibited lower values than HGS, CSO, GA, DE, and OHCRO. A lower standard deviation indicates that Imp-HGS consistently produced more stable and reliable results. It indicates that the optimization process of Imp-HGS generated solutions that were closer to each other, leading to a higher level of consistency in performance. Examining the best values obtained, Imp-HGS achieved either the best or highly competitive results among the algorithms for the majority of the benchmark examples. This indicates that Imp-HGS was successful in finding optimal or near-optimal solutions in the system identification process. Regarding the worst values, Imp-HGS consistently outperformed HGS, GA, DE, and OHCRO, while being comparable to CSO. This suggests that even in worst-case

scenarios, Imp-HGS produced solutions that were relatively close to the optimal or acceptable range, outperforming several other algorithms in this regard.

Taken together, the statistical results demonstrate the significance of Imp-HGS in the IIR system identification benchmark examples. It consistently achieved lower average errors, lower standard deviations, competitive best results, and satisfactory worst results compared to the other algorithms. This indicates that Imp-HGS offers a more reliable, robust, and efficient optimization approach for system identification tasks in comparison to the alternative algorithms.

In terms of computational time, the original version of HGS algorithm has achieved 1.0218 s and 0.7420 s for the same and reduced order cases of Example I, respectively. It achieved 1.4772 s and 1.3327 s for the same and reduced order cases of Example II; 1.8541 s and 1.6338 s for the same and reduced order cases of Example III; lastly, 2.3912 s and 2.1765s for the same and reduced order cases of Example IV, respectively. In the case of the proposed Imp-HGS algorithm, those values were found to be 1.0998 s, 0.7938 s, 1.5717 s, 1.4240 s, 1.9677 s, 1.7386 s, 2.5351 s and 2.3073 s, respectively. As seen, although different mechanisms are embedded within the Imp-HGS algorithm, there is no significant difference between the computational times; signifying efficacy of the proposed Imp-HGS algorithm in terms of computational complexity, as well, for the IIR filter design.

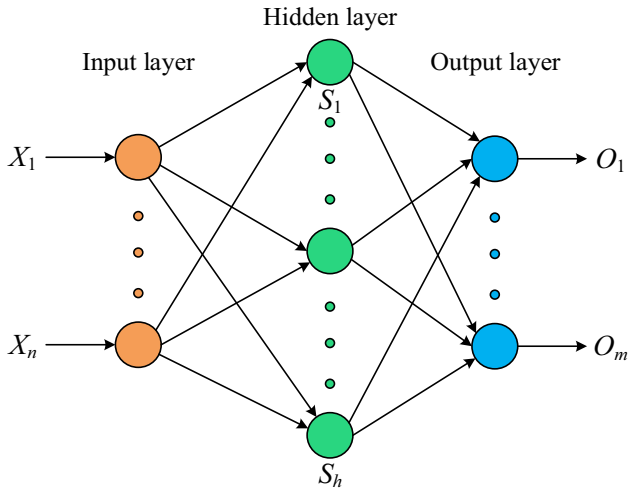


Fig. 7 Structure of MLP neural network

6 Experimental results on multilayer perceptron training

6.1 Multilayer perceptron

Multilayer perceptron (MLP) is a feed-forward neural network where neurons are arranged in a one-directional manner. The network is structured with parallel layers, including the input, hidden, and output layers, where data transition occurs. As shown in Fig. 7, the input layer consists of n nodes, the hidden layer has h nodes, and the output layer has m nodes. The output of MLP is computed in two steps. Firstly, the weighted sums are calculated using $s_j = \sum_{i=1}^{n}(W_{ij}X_i) - \theta_j$ where $J = 1, 2, \ldots, h$, W_{ij} represents the connection weight from the i^{th} node of the input layer to the j^{th} node of the hidden layer, X_i denotes the i^{th} input, and θ_j is the bias of the j^{th} hidden node. Next, each hidden node's output is determined via the sigmoid function,

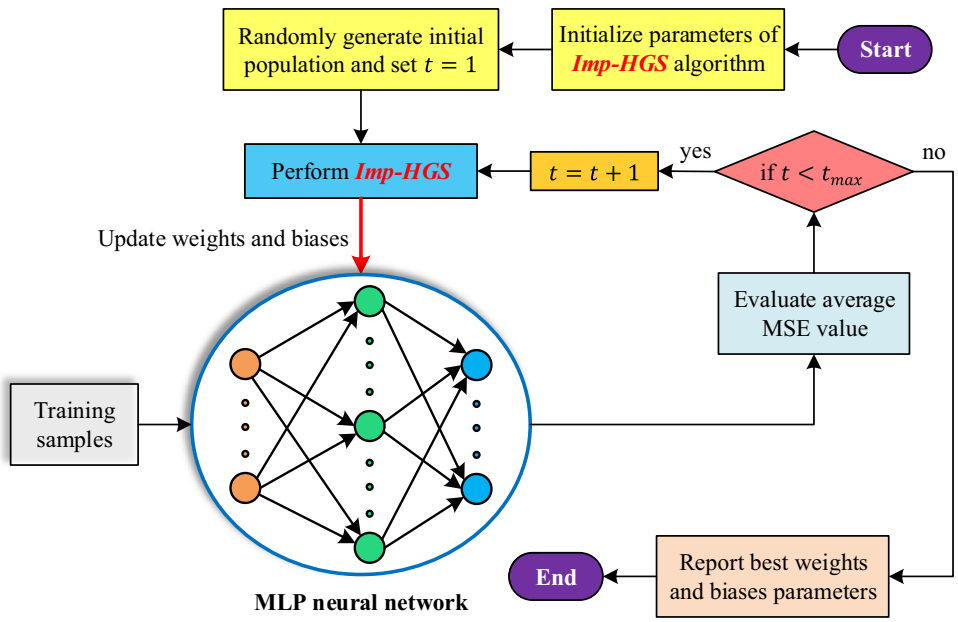


Fig. 8 Imp-HGS algorithm-based MLP trainer

Table 7 Properties of the used datasets

Datasets	Attribute number	Training samples	Test samples	Class number	MLP architecture
Iris	4	150	150	3	4-9-3
XOR	3	8	8	2	3-7-1
Breast cancer	9	599	100	2	9-19-1
Balloon	4	20	20	2	4-9-1
Heart	22	80	187	2	22-45-1
Wine	13	142	36	3	13-27-3
Thyroid	5	172	43	3	5-11-3

which is computed as $S_j = sigmoid(s_j) = 1/(1+e^{-s_j})$ where $j = 1, 2, \ldots, h$ and s_j is the weighted sum calculated in the previous step for the j^{th} hidden node. Once the output of hidden nodes is determined, the final output is computed using $o_k = \sum_{j=1}^{h}(\omega_{jk}S_j) - \theta_k$ and $O_k = sigmoid(o_k) = 1/(1+e^{-o_k})$ where $k = 1, 2, \ldots, m$. The connection weight from the j^{th} hidden node to the k^{th} output node is denoted as ω_{jk}. In order to obtain the desired outputs for defined inputs, it is crucial to train the MLP by finding optimal values for the biases and connection weights. The quality of the MLP's final output is dependent on these factors.

6.2 Experimental setup and analysis of results on classification datasets

The implementation of the proposed Imp-HGS algorithm to MLP training is illustrated in Fig. 8. The classification data sets listed in Table 7 are used to evaluate the performance of the Imp-HGS algorithm for MLP training. The population size and maximum iteration number are selected to be 200 and 250, respectively, and the algorithms are run 30 times. The datasets are classified, and the comparative statistical results are obtained using other recent approaches in literature. In this regard, atom search optimization (ASO) (Eker et al. 2021), dragonfly algorithm (DA) (Gülcü 2022), bat optimization (BAT) (Gülcü 2022) and grey wolf optimizer (GWO) (Mirjalili 2015b) are used alongside the original HGS algorithm in this paper for classification comparisons.

The statistical results obtained from those data sets are displayed in Table 8. Analyzing the average values, it can be observed that Imp-HGS achieved better results across all datasets as it obtained a significantly lower average values compared to HGS, ASO, DA, BAT, and GWO algorithms. Considering the standard deviations, Imp-HGS demonstrated consistent and stable results across the datasets. In the Iris dataset, Imp-HGS achieved a lower standard deviation compared to HGS, ASO, DA, BAT, and GWO, indicating more reliable and consistent optimization outcomes. Similarly, in the XOR dataset, Imp-HGS exhibited a lower standard deviation than HGS, ASO, BAT, and GWO, suggesting a higher level of stability in its performance. In the Heart dataset, Imp-HGS obtained a competitive standard deviation compared to HGS, ASO, and BAT, while outperforming

Table 8 Statistical results obtained from different data sets

Datasets	Measure	Imp-HGS	HGS	ASO	DA	BAT	GWO
Iris	Average	**9.50E−03**	1.46E−02	1.83E−02	8.46E−02	1.95E−01	2.29E−02
	Standard deviation	1.47E−03	2.20E−03	2.77E−03	1.07E−01	1.57E−01	3.20E−03
XOR	Average	**5.80E−07**	1.96E−06	5.72E−03	4.95E−02	1.35E−01	9.41E−03
	Standard deviation	2.20E−06	6.24E−06	2.37E−02	6.03E−02	5.97E−02	2.95E−02
Breast cancer	Average	**9.78E−04**	1.10E−03	3.47E−03	6.03E−03	6.68E−03	1.20E−03
	Standard deviation	3.01E−05	6.63E−05	1.64E−03	3.29E−03	7.35E−03	7.45E−05
Balloon	Average	**2.13E−37**	1.67E−34	3.47E−08	3.35E−06	6.12E−03	9.38E−15
	Standard deviation	5.73E−36	9.00E−34	1.53E−07	7.56E−06	2.05E−02	2.81E−14
Heart	Average	**6.32E−02**	6.69E−02	9.52E−02	2.24E−01	1.51E−01	1.23E−01
	Standard deviation	1.73E−02	1.36E−02	1.38E−02	1.41E−02	2.98E−02	7.70E−03
Wine	Average	**5.50E−02**	1.28E−01	NR	1.85E−01	3.99E−01	NR
	Standard deviation	1.46E−01	1.31E−01	NR	1.20E−01	1.84E−01	NR
Thyroid	Average	**3.98E−02**	1.22E−01	NR	9.59E−02	1.94E−01	NR
	Standard deviation	3.49E−02	3.02E−02	NR	6.65E−02	9.76E−02	NR

NR not reported

Table 9 Average classification rate (%)

Datasets	Imp-HGS	HGS	ASO	DA	BAT	GWO
Iris	98.85	98.67	89.33	93.40	84.56	91.33
XOR	100.00	100.00	100.00	93.33	84.17	100.00
Breast cancer	100.00	100.00	99.00	95.80	96.20	99.00
Balloon	100.00	100.00	100.00	100.00	99.33	100.00
Heart	94.15	92.50	73.75	70.30	69.75	75.00
Wine	96.00	94.38	NR	88.80	72.50	NR
Thyroid	86.35	84.65	NR	91.40	85.43	NR

NR not reported

GWO. In summary, based on the statistical results, Imp-HGS demonstrates promising performance in the employed datasets. It achieved lower average values and exhibited more stable results, as indicated by lower standard deviations, in comparison to HGS, ASO, DA, BAT, and GWO in various datasets. Therefore, the results indicate the better performance of the Imp-HGS algorithm for MLP training.

Table 9 provides the average classification rates obtained using different algorithms. This table provides insights into the performance of the Imp-HGS, HGS, ASO, DA, BAT, and GWO algorithms for different datasets. By analyzing the results, we can observe that the proposed Imp-HGS algorithm consistently demonstrates better performance compared to the other algorithms in several datasets. In the Iris dataset, the Imp-HGS algorithm achieved an impressive average classification rate of 98.85%, outperforming HGS, ASO, DA, BAT, and GWO. This suggests that the Imp-HGS algorithm is effective in accurately classifying Iris flowers. For the XOR dataset, the Imp-HGS algorithm achieved a perfect classification rate of 100%, demonstrating its ability to accurately solve the XOR problem. In contrast, the other algorithms had lower classification rates, indicating the superiority of Imp-HGS in this particular dataset. In the Breast Cancer dataset, the Imp-HGS algorithm achieved a perfect average classification rate of 100%, surpassing ASO, DA, BAT, and GWO. This indicates that Imp-HGS is highly effective in classifying breast cancer cases, potentially aiding in accurate diagnosis. Similarly, in the Balloon dataset, the Imp-HGS algorithm achieved a perfect average classification rate of 100%, outperforming all other algorithms. This suggests that Imp-HGS is well-suited for accurately classifying instances in the Balloon dataset. In the Heart dataset, although the Imp-HGS algorithm obtained a lower average classification rate of 94.15%. It outperformed HGS, ASO, DA, BAT, and GWO, indicating its effectiveness in heart disease classification. While the Wine dataset does not provide the classification rates for ASO and GWO, Imp-HGS achieved a higher average classification rate (96%) compared to HGS, indicating its superior performance in wine classification. In the Thyroid dataset, the Imp-HGS algorithm achieved an average classification rate of 86.35%, surpassing HGS, DA and BA. This suggests that Imp-HGS is effective in accurately classifying thyroid instances. Overall, the Imp-HGS algorithm consistently demonstrates better performance in terms of average classification rates across various datasets, showcasing its effectiveness in solving classification problems. Its superior performance can be attributed to its optimization capabilities and ability to find optimal solutions for the classification tasks at hand. Therefore, the proposed

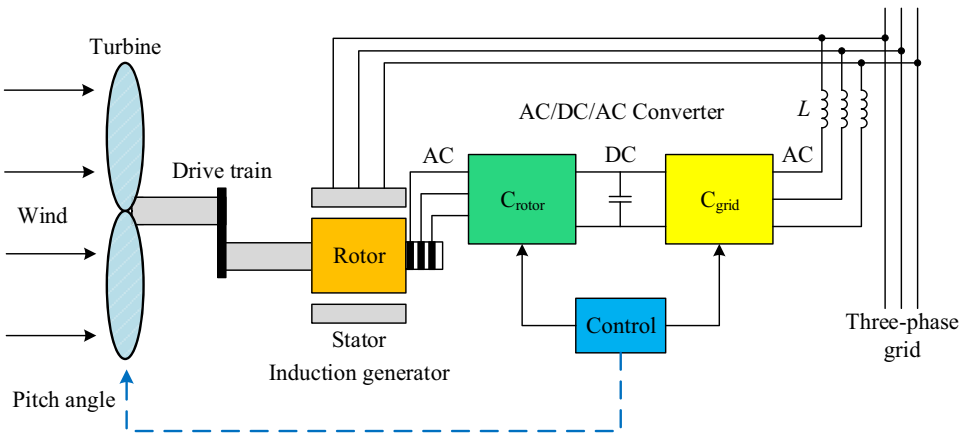


Fig. 9 DFIG-based wind turbine system

Imp-HGS algorithm also behaves as a good tool for MLP training purposes.

In terms of computational time, the original HGS algorithm achieved 826.8312 s, 2.2250 s, 2734.0174 s, 34.4750 s, 983.1200 s, 936.2310 s and 912.4155 s for Iris, XOR, Breast cancer, Balloon, Heart, Wine and Thyroid datasets, respectively. Those values were obtained as 892.1974s, 2.5813 s, 3072.2021s, 37.5413 s, 1102.3558 s, 1068.3757 s and 1029.6366 s, respectively, by the proposed Imp-HGS algorithm. As seen, although different mechanisms are embedded within the Imp-HGS algorithm, there is no significant difference between the computational times; signifying efficacy of the proposed Imp-HGS algorithm in terms of computational complexity, as well, for the MLP training.

7 Experimental results on controller design

7.1 Doubly fed induction generator-based wind turbine system

Figure 9 showcases a simplified yet powerful illustration of the cutting-edge doubly fed induction generator (DFIG)-based wind turbine system, representing a remarkable breakthrough in renewable energy. Comprising the wind turbine, drive train, induction generator, AC/DC/AC converters, and power transformer, this state-of-the-art system harnesses the wind's potential energy in two stages. First, wind power is ingeniously converted into mechanical power, converted into clean, efficient, and sustainable electrical power (Nasef et al. 2022).

The DFIG-based wind turbine system provides a versatile solution to harness wind energy. As shown in Fig. 9, this system comprises a wind turbine, drive train, induction generator, AC/DC/AC converters, and a power transformer. This highly efficient system converts wind power to mechanical power, subsequently converted to electrical power. To achieve variable speed, the rotor circuitry of the DFIG is electrically controlled, and two operating modes exist (i) where the rotor speed is greater than synchronous speed and (ii) where the opposite is true. In mode (i), a super-synchronous operation occurs, where the rotor and stator winding communicate electrical power for the grid. In contrast, in mode (ii), a sub-synchronous operation occurs, where the stator winding supplies power together with the grid and the rotor winding (Bounar et al. 2019).

7.2 Experimental setup and analysis of results on PID controller design

The optimal controller design for a doubly fed induction generator-based wind energy conversion system was achieved through a combination of advanced optimization algorithms. In particular, the Imp-HGS algorithm has been utilized in conjunction with the reptile search algorithm (RSA) (Izci et al. 2022), gravitational search algorithm (GSA) (Bharti et al. 2021), particle swarm optimization (PSO) (Bharti et al. 2021), and bacterial foraging optimization (BFO) (Bharti et al. 2021). These algorithms were applied to the 6*th* order transfer function model, as presented in Eq. (9), to study the transient behavior of the system.

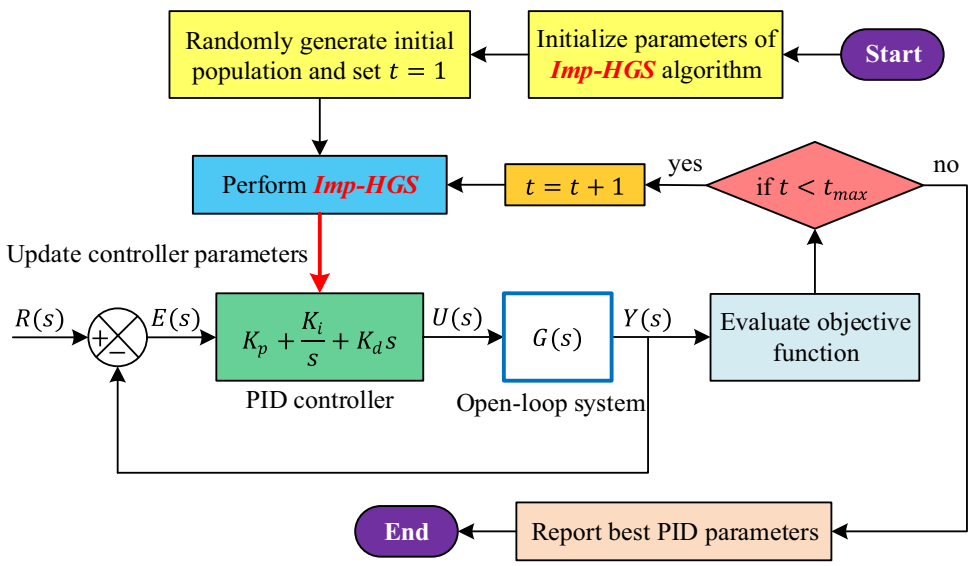


Fig. 10 Imp-HGS algorithm-based PID controller design

Table 10 Closed-loop transfer functions obtained for different algorithms

Algorithm	Closed-loop transfer function
Imp-HGS	$\frac{3.095\times10^{-5}s^8-0.1618s^7-254.7s^6+7.155\times10^5s^5+8.458\times10^8s^4+6.019\times10^{11}s^3+1.041\times10^{14}s^2+4.333\times10^{15}s+3.282\times10^{16}}{3.095\times10^{-5}s^8+0.8382s^7+2085s^6+9.385\times10^6s^5+5.636\times10^9s^4+3.302\times10^{12}s^3+2.311\times10^{14}s^2+5.293\times10^{15}s+3.282\times10^{16}}$
HGS	$\frac{3.833\times10^{-5}s^8-0.2022s^7-306.1s^6+8.989\times10^5s^5+1.005\times10^9s^4+7.05\times10^{11}s^3+1.017\times10^{14}s^2+3.977\times10^{15}s+3.105\times10^{16}}{3.833\times10^{-5}s^8+0.7978s^7+2034s^6+9.569\times10^6s^5+5.795\times10^9s^4+3.405\times10^{12}s^3+2.287\times10^{14}s^2+4.937\times10^{15}s+3.105\times10^{16}}$
RSA	$\frac{2.83\times10^{-5}s^8-0.1468s^7-239.3s^6+6.454\times10^5s^5+8.024\times10^8s^4+5.789\times10^{11}s^3+1.143\times10^{14}s^2+5.218\times10^{15}s+4.962\times10^{16}}{2.83\times10^{-5}s^8+0.8532s^7+2101s^6+9.315\times10^6s^5+5.592\times10^9s^4+3.279\times10^{12}s^3+2.413\times10^{14}s^2+6.178\times10^{15}s+4.962\times10^{16}}$
GSA	$\frac{3.859\times10^{-5}s^8-0.2029s^7-311.4s^6+9.004\times10^5s^5+1.027\times10^9s^4+7.245\times10^{11}s^3+1.125\times10^{14}s^2+4.905\times10^{15}s+5.284\times10^{16}}{3.859\times10^{-5}s^8+0.7971s^7+2029s^6+9.57\times10^6s^5+5.817\times10^9s^4+3.424\times10^{12}s^3+2.395\times10^{14}s^2+5.865\times10^{15}s+5.284\times10^{16}}$
BFO	$\frac{3.852\times10^{-5}s^8-0.2032s^7-307.4s^6+9.035\times10^5s^5+1.009\times10^9s^4+7.084\times10^{11}s^3+1.024\times10^{14}s^2+4.459\times10^{15}s+5.241\times10^{16}}{3.852\times10^{-5}s^8+0.7968s^7+2033s^6+9.574\times10^6s^5+5.799\times10^9s^4+3.408\times10^{12}s^3+2.294\times10^{14}s^2+5.419\times10^{15}s+5.241\times10^{16}}$
PSO	$\frac{3.208\times10^{-6}s^8-0.0153s^7-34.31s^6+6.325\times10^4s^5+1.233\times10^8s^4+9.717\times10^{10}s^3+3.419\times10^{13}s^2+1.918\times10^{15}s+2.431\times10^{16}}{3.208\times10^{-6}s^8+0.9847s^7+2306s^6+8.733\times10^6s^5+4.913\times10^9s^4+2.797\times10^{12}s^3+1.612\times10^{14}s^2+2.878\times10^{15}s+2.431\times10^{16}}$

Table 11 Controller parameters obtained from performing different algorithms and the obtained transient response specifications

PID design		Imp-HGS	HGS	RSA	GSA	BFO	PSO
Controller parameter	K_p	16.4211	14.9782	18.7151	16.9400	14.9400	6.2392
	K_i	150.5726	142.4485	227.6068	242.4000	240.4000	111.4924
	K_d	0.09552	0.1183	0.08735	0.1191	0.1189	0.0099
Transient response specification	$OS(\%)$	0	0.4898	2.7713	4.6206	6.1632	7.6927
	t_r(s)	0	0	0	0	0	0
	t_s(s)	0.1131	0.1203	0.1570	0.1970	0.2158	0.3338
	t_p(s)	0	0.2264	0.1317	0.1330	0.1355	0.2024

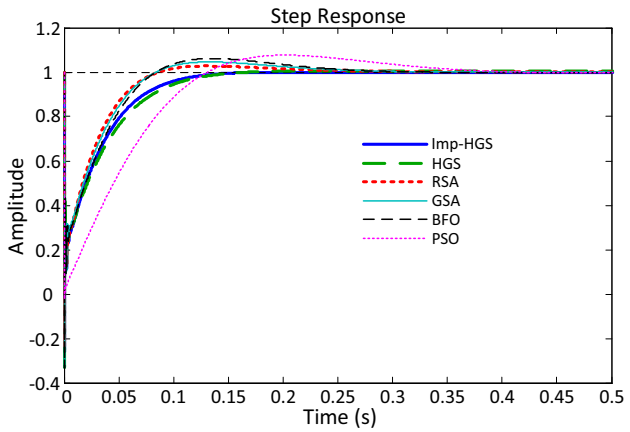


Fig. 11 Step response obtained from different approaches

$$G(s)=\frac{0.000324s^6-1.75s^5-2366s^4+7.9\times10^6s^3+7.5\times10^9s^2+5\times10^{12}s+2.18\times10^{14}}{s^6+2340s^5+8.67\times10^6s^4+4.79\times10^9s^3+2.7\times10^{12}s^2+1.27\times10^{14}s+9.6\times10^{14}} \tag{9}$$

A detailed discussion of this transfer function model can be found in the work of Ko et al. (2008). The step response of this model yields a percent overshoot $\%OS=0$, a rise time $t_r=0.2359$ s, a settling time $t_s=0.4197$ s, and a peak time $t_p=0.7780$ s. However, in this study, we aimed to further enhance these values by applying the proposed Imp-HGS algorithm-based PID controller. The detailed implementation procedure of the Imp-HGS algorithm-based PID controller design for the DFIG-based wind turbine system is provided in Fig. 10.

The PID controller has the following form where the gains known as proportional, integral, and derivative are denoted by K_p, K_i, and K_d, respectively (Ekinci et al. 2022).

$$C(s)=K_p+\frac{K_i}{s}+K_ds \tag{10}$$

To have an efficient optimization for the gains provided in Eq. (10), the F_{ZLG} cost function given in Eq. (11) has been utilized in this work (Ekinci et al. 2022).

$$F_{ZLG}=(1-e^{-\rho})\left(\frac{\%OS}{100}+e_{ss}\right)+e^{-\rho}(t_s-t_r) \tag{11}$$

Here, the balancing coefficient was set to $\rho=1$ as it was determined to be the most suitable value. For the optimization task, the boundaries for the gains were set as $10^{-1}\le K_p\le 20, 1\le K_i\le 250$, and $10^{-3}\le K_d\le 1$. Besides, the total iteration number was set to 50 using a population size of 30. Each algorithm was run 30 times to obtain the results. Table 10 presents the obtained closed-loop transfer functions obtained from performing different algorithms. Those transfer functions have been formulated by using the

gain parameters obtained from performing different algorithms (see Table 11 for those values) and implementing the procedure explained in Fig. 10 for each approach. The transient response specifications demonstrated in Table 11 and the step response shown in Fig. 11 illustrate the more excellent feature of the proposed method.

Comparing the results of the Imp-HGS algorithm with HGS, RSA, GSA, BFO, and PSO algorithms, it is evident that the Imp-HGS algorithm exhibits better performance in several aspects. When considering the percent overshoot, the Imp-HGS algorithm achieves a value of 0%, indicating that it effectively eliminates any overshoot in the system response. On the other hand, HGS shows a percent overshoot of 0.4898, while RSA, GSA, BFO, and PSO algorithms have even higher values. This suggests that the Imp-HGS algorithm provides a more stable response with minimal oscillations, ensuring better control of the wind turbine system. In terms of rise time, settling time, and peak time, the Imp-HGS algorithm achieves superior performance by attaining values of 0 for rise and peak times and the lowest value (0.1131 s) for settling time. This implies that the Imp-HGS algorithm provides the fastest response with no significant delays or deviations from the desired output. In contrast, the other algorithms (HGS, RSA, GSA, BFO, and PSO) exhibit non-zero values for peak time and higher settling times, indicating comparatively slower response times and longer settling times.

The better performance of the Imp-HGS algorithm in terms of these transient response specifications can be attributed to its optimization capabilities. The algorithm is able to fine-tune the PID controller parameters efficiently, ensuring optimal system performance and minimizing transient effects such as overshoot, rise time, settling time, and peak time. Overall, the Imp-HGS algorithm demonstrates better control over the wind turbine system's transient response compared to the other algorithms considered. Its ability to eliminate overshoot, achieve faster rise time, shorter settling time, and minimal peak time showcases its effectiveness in achieving stable and efficient operation of the wind turbine system.

In addition, the original HGS algorithm achieved a computational time of 25.7423 s while the proposed Imp-HGS algorithm achieved 28.5960 s for the PID design. Similar to other engineering design cases, although different mechanisms are embedded within the Imp-HGS algorithm, there is no significant difference between the computational times; signifying efficacy of the proposed Imp-HGS algorithm in terms of computational complexity, as well, for the PID design.

8 Conclusion and future works

Based on the findings of this study, it can be concluded that the Imp-HGS algorithm represents a significant advancement in the field of metaheuristic optimization. By integrating PS and elite OBL mechanisms, the algorithm successfully addresses the limitations of original HGS algorithm and demonstrates superior performance in solving complex optimization problems with non-differentiable nature and many decision variables. The empirical evaluation using the CEC2019 and CEC2020 test suites, as well as diverse engineering problems such as IIR model identification, MLP training, and PID controller design for a DFIG-based wind turbine system, consistently showcases the algorithm's effectiveness and outperformance compared to state-of-the-art algorithms. The contributions of this work can be summarized twofold. Firstly, the introduction of the Imp-HGS algorithm provides a valuable tool for various engineering applications, surpassing the limitations of traditional optimization approaches and demonstrating improved performance in solving complex optimization problems compared to state-of-the-art reported approaches. Secondly, the extensive experiments conducted on diverse engineering problems provide empirical evidence of the algorithm's capabilities, highlighting its potential to revolutionize optimization and decision-making processes.

Looking towards the future, further research can be conducted to enhance the algorithm's performance and adaptability. This may involve investigating additional pattern search strategies and intelligent learning mechanisms, as well as exploring hybridization with other metaheuristic techniques. Real-world applications and benchmarking against existing optimization methods can provide deeper insights into the algorithm's practical applicability and identify areas for improvement. Additionally, comprehensive parameter tuning, sensitivity analysis, and scalability considerations are essential for unlocking the algorithm's full potential. In conclusion, the Imp-HGS algorithm presents a promising and valuable solution for solving a wide range of complex engineering optimization problems. Its enhanced performance, demonstrated through empirical evaluations, positions it as a significant advancement in the field of metaheuristic optimization. Further research and practical applications are encouraged to fully explore the algorithm's capabilities and contribute to the ongoing development of efficient optimization techniques.

Funding The authors received no specific funding for this study.

Data availability Data is available from the authors upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflicts of interest to report regarding the present study.

References

Abualigah L, Diabat A, Mirjalili S, Abd Elaziz M, Gandomi AH (2021) The arithmetic optimization algorithm. Comput Methods Appl Mech Eng 376:113609. https://doi.org/10.1016/j.cma.2020.113609

AbuShanab WS, Abd Elaziz M, Ghandourah EI, Moustafa EB, Elsheikh AH (2021) A new fine-tuned random vector functional link model using Hunger games search optimizer for modeling friction stir welding process of polymeric materials. J Market Res 14:1482–1493. https://doi.org/10.1016/j.jmrt.2021.07.031

Agahian S, Akan T (2022) Battle royale optimizer for training multi-layer perceptron. Evol Syst 13:563–575. https://doi.org/10.1007/s12530-021-09401-5

Agushaka JO, Ezugwu AE, Abualigah L (2022) Dwarf mongoose optimization algorithm. Comput Methods Appl Mech Eng 391:114570. https://doi.org/10.1016/j.cma.2022.114570

Agushaka JO, Ezugwu AE, Abualigah L (2022b) Gazelle optimization algorithm: a novel nature-inspired metaheuristic optimizer. Neural Comput Appl. https://doi.org/10.1007/s00521-022-07854-6

Ahmadianfar I, Heidari AA, Gandomi AH, Chu X, Chen H (2021) RUN beyond the metaphor: an efficient optimization algorithm based on Runge Kutta method. Expert Syst Appl 181:115079. https://doi.org/10.1016/j.eswa.2021.115079

Ahmed HA, Kamel S, Korashy A, Jurado F (2019) Application of Harris Hawks algorithm for frequency response enhancement of two-area interconnected power system with DFIG based wind turbine. In: 2019 21st international middle east power systems conference (MEPCON). pp 568–574. IEEE

Bacanin N, Alhazmi K, Zivkovic M, Venkatachalam K, Bezdan T, Nebhen J (2022) Training Multi-layer perceptron with enhanced brain storm optimization metaheuristics. Comput Mater Contin 70:4199–4215. https://doi.org/10.32604/cmc.2022.020449

Bansal P, Gupta S, Kumar S, Sharma S, Sharma S (2019) MLP-LOA: a metaheuristic approach to design an optimal multilayer perceptron. Soft Comput 23:12331–12345. https://doi.org/10.1007/s00500-019-03773-2

Benamor A, Benchouia MT, Srairi K, Benbouzid MEH (2019) A new rooted tree optimization algorithm for indirect power control of wind turbine based on a doubly-fed induction generator. ISA Trans 88:296–306. https://doi.org/10.1016/j.isatra.2018.11.023

Bharti OP, Saket RK, Nagar SK (2017) Controller design for doubly fed induction generator using particle swarm optimization technique. Renew Energy 114:1394–1406. https://doi.org/10.1016/j.renene.2017.06.061

Bharti OP, Sarita K, Vardhan ASS, Vardhan ASS, Saket RK (2021) Controller design for DFIG-based WT using gravitational search algorithm for wind power generation. IET Renew Power Gener 15:1956–1967. https://doi.org/10.1049/rpg2.12118

Bounar N, Labdai S, Boulkroune A (2019) PSO–GSA based fuzzy sliding mode controller for DFIG-based wind turbine. ISA Trans 85:177–188. https://doi.org/10.1016/j.isatra.2018.10.020

Boureguig K, Soued S, Ouagueni F, Chahmi A (2023) Optimal metaheuristic-based feedback linearization control of DFIG wind turbine system. J Electr Eng Technol. https://doi.org/10.1007/s42835-023-01386-2

Braik M, Sheta A, Al-Hiary H (2021) A novel meta-heuristic search algorithm for solving optimization problems: capuchin search algorithm. Neural Comput Appl 33:2515–2547. https://doi.org/10.1007/s00521-020-05145-6

Carbas S, Toktas A, Ustun D (2021) Introduction and overview: nature-inspired metaheuristic algorithms for engineering optimization applications

Chakraborty S, Kumar Saha A, Sharma S, Mirjalili S, Chakraborty R (2021) A novel enhanced whale optimization algorithm for global optimization. Comput Ind Eng 153:107086. https://doi.org/10.1016/j.cie.2020.107086

Chakraborty S, Saha AK, Chakraborty R, Saha M, Nama S (2022) HSWOA: an ensemble of hunger games search and whale optimization algorithm for global optimization. Int J Intell Syst 37:52–104. https://doi.org/10.1002/int.22617

Cuevas E, Avalos O, Gálvez J (2023) IIR system identification using several optimization techniques: a review analysis

Durmuş B (2022) Infinite impulse response system identification using average differential evolution algorithm with local search. Neural Comput Appl 34:375–390. https://doi.org/10.1007/s00521-021-06399-4

Eker E, Kayri M, Ekinci S, Izci D (2021) A new fusion of ASO with SA algorithm and its applications to MLP training and DC motor speed control. Arab J Sci Eng 46:3889–3911. https://doi.org/10.1007/s13369-020-05228-5

Ekinci S, Izci D, Al Nasar MR, Abu Zitar R, Abualigah L (2022) Logarithmic spiral search based arithmetic optimization algorithm with selective mechanism and its application to functional electrical stimulation system control. Soft Comput 26:12257–12269. https://doi.org/10.1007/s00500-022-07068-x

Ekinci S, Izci D, Abu Zitar R, Alsoud AR, Abualigah L (2022) Development of Lévy flight-based reptile search algorithm with local search ability for power systems engineering design problems. Neural Comput Appl 34:20263–20283. https://doi.org/10.1007/s00521-022-07575-w

Ekinci S, Izci D, Eker E, Abualigah L (2023a) An effective control design approach based on novel enhanced aquila optimizer for automatic voltage regulator. Artif Intell Rev 56:1731–1762. https://doi.org/10.1007/s10462-022-10216-2

Ekinci S, Izci D, Abualigah L (2023b) A novel balanced Aquila optimizer using random learning and Nelder–Mead simplex search mechanisms for air–fuel ratio system control. J Braz Soc Mech Sci Eng 45:68. https://doi.org/10.1007/s40430-022-04008-6

Ezugwu AE, Agushaka JO, Abualigah L, Mirjalili S, Gandomi AH (2022) Prairie dog optimization algorithm. Neural Comput Appl 34:20017–20065. https://doi.org/10.1007/s00521-022-07530-9

Gharehchopogh FS (2023) Quantum-inspired metaheuristic algorithms: comprehensive survey and classification. Artif Intell Rev 56:5479–5543. https://doi.org/10.1007/s10462-022-10280-8

Gharehchopogh FS, Ucan A, Ibrikci T, Arasteh B, Isik G (2023) Slime mould algorithm: a comprehensive survey of its variants and applications. Arch Comput Methods Eng 30:2683–2723. https://doi.org/10.1007/s11831-023-09883-3

Gülcü Ş (2022) Training of the feed forward artificial neural networks using dragonfly algorithm. Appl Soft Comput 124:109023. https://doi.org/10.1016/j.asoc.2022.109023

Gupta S, Deep K (2020) A novel hybrid sine cosine algorithm for global optimization and its application to train multilayer perceptrons. Appl Intell 50:993–1026. https://doi.org/10.1007/s10489-019-01570-w

Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Futur Gener Comput Syst 97:849–872. https://doi.org/10.1016/j.future.2019.02.028

Heidari AA, Faris H, Mirjalili S, Aljarah I, Mafarja M (2020) Ant lion optimizer: theory, literature review, and application in multi-layer perceptron neural networks. In: Mirjalili S, Song Dong J, Lewis A (eds) Studies in Computational Intelligence. Springer International Publishing, Cham, pp 23–46

Hong H, Tsangaratos P, Ilia I, Loupasakis C, Wang Y (2020) Introducing a novel multi-layer perceptron network based on stochastic gradient descent optimized by a meta-heuristic algorithm for landslide susceptibility mapping. Sci Total Environ 742:140549. https://doi.org/10.1016/j.scitotenv.2020.140549

Huang L, Yang B, Zhang X, Yin L, Yu T, Fang Z (2019) Optimal power tracking of doubly fed induction generator-based wind turbine using swarm moth–flame optimizer. Trans Inst Meas Control 41:1491–1503. https://doi.org/10.1177/0142331217712091

Hussien AG, Abualigah L, Abu Zitar R, Hashim FA, Amin M, Saber A, Almotairi KH, Gandomi AH (2022) Recent advances in Harris Hawks optimization: a comparative study and applications. Electronics (Basel) 11:1919. https://doi.org/10.3390/electronics11121919

Irmak B, Karakoyun M, Gülcü Ş (2022) An improved butterfly optimization algorithm for training the feed-forward artificial neural networks. Soft Comput. https://doi.org/10.1007/s00500-022-07592-w

Izci D (2022) A novel modified arithmetic optimization algorithm for power system stabilizer design. Sigma J Eng Nat Sci 40:529–541. https://doi.org/10.14744/sigma.2022.00056

Izci D, Ekinci S (2021) Comparative performance analysis of slime mould algorithm for efficient design of proportional–integral–derivative controller. Electrica 21:151–159. https://doi.org/10.5152/electrica.2021.20077

Izci D, Ekinci S, Eker E, Demirören A (2022) Multi-strategy modified INFO algorithm: performance analysis and application to functional electrical stimulation system. J Comput Sci 64:101836. https://doi.org/10.1016/j.jocs.2022.101836

Izci D, Ekinci S (2023) A novel-enhanced metaheuristic algorithm for FOPID-controlled and Bode's ideal transfer function–based buck converter system. Trans Inst Meas Control 45:1854–1872. https://doi.org/10.1177/01423312221140671

Izci D, Ekinci S, Eker E, Kayri M (2022c) Augmented hunger games search algorithm using logarithmic spiral opposition-based learning for function optimization and controller design. J King Saud Univ Eng Sci. https://doi.org/10.1016/j.jksues.2022.03.001

Izci D, Ekinci S, Hussien AG (2023) Effective PID controller design using a novel hybrid algorithm for high order systems. PLoS ONE 18:e0286060. https://doi.org/10.1371/journal.pone.0286060

Izci D, Ekinci S, Eker E, Demirören A (2023b) Biomedical application of a random learning and elite opposition-based weighted mean of vectors algorithm with pattern search mechanism. J Control Autom Electr Syst 34:333–343. https://doi.org/10.1007/s40313-022-00959-2

Izci D, Ekinci S (2022) A novel improved version of hunger games search algorithm for function optimization and efficient controller design of buck converter system. e-Prime Adv Electr Eng Electron Energy 2:100039. https://doi.org/10.1016/j.prime.2022.100039

Izci D, Ekinci S, Budak C, Gider V (2022) PID controller design for DFIG-based wind turbine via reptile search algorithm. In: 2022b Global Energy Conference (GEC), pp 154–158. IEEE

Izci D, Ekinci S, Eker E, Kayri M (2022d) A novel modified opposition-based hunger games search algorithm to design fractional order proportional-integral-derivative controller for magnetic ball suspension system. Adv Control Appl **4**:e96. https://doi.org/10.1002/adc2.96

Karaboga D, Akay B (2009) A comparative study of artificial Bee colony algorithm. Appl Math Comput 214:108–132. https://doi.org/10.1016/j.amc.2009.03.090

Karad SG, Thakur R (2022) Enhanced control of doubly fed induction generator based wind turbine system using soft computing assisted fractional order controller. Renew Energy Focus 43:291–308. https://doi.org/10.1016/j.ref.2022.10.006

Ko H-S, Yoon G-G, Kyung N-H, Hong W-P (2008) Modeling and control of DFIG-based variable-speed wind-turbine. Electric Power Syst Res 78:1841–1849. https://doi.org/10.1016/j.epsr.2008.02.018

Kumar M, Aggarwal A, Rawat TK (2016) Bat algorithm: application to adaptive infinite impulse response system identification. Arab J Sci Eng 41:3587–3604. https://doi.org/10.1007/s13369-016-2222-3

Kutlu Onay F, Aydemir SB (2022) Chaotic hunger games search optimization algorithm for global optimization and engineering problems. Math Comput Simul 192:514–536. https://doi.org/10.1016/j.matcom.2021.09.014

Labdai S, Bounar N, Boulkroune A, Hemici B, Nezli L (2022) Artificial neural network-based adaptive control for a DFIG-based WECS. ISA Trans 128:171–180. https://doi.org/10.1016/j.isatra.2021.11.045

Lee EH (2023) Inflow prediction of centralized reservoir for the operation of pump station in urban drainage systems using improved multilayer perceptron using existing optimizers combined with metaheuristic optimization algorithms. Water (Basel) 15:1543. https://doi.org/10.3390/w15081543

Lee WJ, Lee EH (2022) Runoff prediction based on the discharge of pump stations in an urban stream using a modified multi-layer perceptron combined with meta-heuristic optimization. Water (Basel) 14:99. https://doi.org/10.3390/w14010099

Li S, Li X, Chen H, Zhao Y, Dong J (2021) A novel hybrid hunger games search algorithm with differential evolution for improving the behaviors of non-cooperative animals. IEEE Access 9:164188–164205. https://doi.org/10.1109/ACCESS.2021.3132617

Li X-D, Wang J-S, Hao W-K, Wang M, Zhang M (2022) Multilayer perceptron classification method of medical data based on biogeography-based optimization algorithm with probability distributions. Appl Soft Comput 121:108766. https://doi.org/10.1016/j.asoc.2022.108766

Luo Q, Ling Y, Zhou Y (2020) Modified whale optimization algorithm for infinitive impulse response system identification. Arab J Sci Eng 45:2163–2176. https://doi.org/10.1007/s13369-019-04093-1

Ma, B.J.: Hybrid adaptive moth-flame optimizer and opposition-based learning for training multilayer perceptrons. Presented at the (2022)

Ma BJ, Liu S, Heidari AA (2022) Multi-strategy ensemble binary hunger games search for feature selection. Knowl Based Syst 248:108787. https://doi.org/10.1016/j.knosys.2022.108787

Mirjalili S (2015a) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Syst 89:228–249. https://doi.org/10.1016/j.knosys.2015.07.006

Mirjalili S (2015b) How effective is the Grey Wolf optimizer in training multi-layer perceptrons. Appl Intell 43:150–161. https://doi.org/10.1007/s10489-014-0645-7

Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and

multi-objective problems. Neural Comput Appl 27:1053–1073. https://doi.org/10.1007/s00521-015-1920-1

Moghanian S, Saravi FB, Javidi G, Sheybani EO (2020) GOAMLP: network intrusion detection with multilayer perceptron and grasshopper optimization algorithm. IEEE Access 8:215202–215213. https://doi.org/10.1109/ACCESS.2020.3040740

Mohammadi A, Zahiri SH, Razavi SM (2019) Infinite impulse response systems modeling by artificial intelligent optimization methods. Evol Syst 10:221–237. https://doi.org/10.1007/s12530-018-9218-z

Mohammadi A, Zahiri SH, Razavi SM, Suganthan PN (2021) Design and modeling of adaptive IIR filtering systems using a weighted sum - variable length particle swarm optimization. Appl Soft Comput 109:107529. https://doi.org/10.1016/j.asoc.2021.107529

Mohammadi A, Sheikholeslam F, Mirjalili S (2022) Inclined planes system optimization: theory, literature review, and state-of-the-art versions for IIR system identification. Expert Syst Appl 200:117127. https://doi.org/10.1016/j.eswa.2022.117127

Mohammadzadeh H, Gharehchopogh FS (2021) A multi-agent system based for solving high-dimensional optimization problems: a case study on email spam detection. Int J Commun Syst 34:1. https://doi.org/10.1002/dac.4670

Mohammadzadeh H, Gharehchopogh FS (2021a) Feature selection with binary symbiotic organisms search algorithm for email spam detection. Int J Inf Technol Decis Mak 20:469–515. https://doi.org/10.1142/S0219622020500546

Mostafa MA, El-Hay EA, ELkholy MM (2023) Optimal maximum power point tracking of wind turbine doubly fed induction generator based on driving training algorithm. Wind Eng 47:671–687. https://doi.org/10.1177/0309524X221150443

Muisyo IN, Muriithi CM, Kamau SI (2022) Enhancing low voltage ride through capability of grid connected DFIG based WECS using WCA-PSO tuned STATCOM controller. Heliyon 8:e09999. https://doi.org/10.1016/j.heliyon.2022.e09999

Nasef SA, Hassan AA, Elsayed HT, Zahran MB, El-Shaer MK, Abdelaziz AY (2022) Optimal tuning of a new multi-input multi-output fuzzy controller for doubly fed induction generator-based wind energy conversion system. Arab J Sci Eng 47:3001–3021. https://doi.org/10.1007/s13369-021-05946-4

Nguyen H, Bui X-N (2021) A novel hunger games search optimization-based artificial neural network for predicting ground vibration intensity induced by mine blasting. Nat Resour Res. https://doi.org/10.1007/s11053-021-09903-8

Niu Y, Yan X, Wang Y, Niu Y (2022) Dynamic opposite learning enhanced artificial ecosystem optimizer for IIR system identification. J Supercomput 78:13040–13085. https://doi.org/10.1007/s11227-022-04367-w

Özmen H, Ekinci S, Izci D (2023) Boosted arithmetic optimization algorithm with elite opposition-based pattern search mechanism and its promise to design microstrip patch antenna for WLAN and WiMAX. Int J Model Simul 1–16. https://doi.org/10.1080/02286203.2023.2196736

Palanimuthu K, Kim HS, Joo YH (2022) T-S fuzzy sliding mode control for double-fed induction generator-based wind energy system with a membership function-dependent H∞-approach. Inf Sci (n y) 596:73–92. https://doi.org/10.1016/j.ins.2022.03.005

Panda G, Pradhan PM, Majhi B (2011) IIR system identification using cat swarm optimization. Expert Syst Appl 38:12671–12683. https://doi.org/10.1016/j.eswa.2011.04.054

El Qouarti O, Essadki A, Laghridat H, Nasser T (2023) Power optimisation of DFIG based WECS using SMC and metaheuristic algorithms. Presented at the (2023)

Sharifi MA, Mojallali H (2015) A modified imperialist competitive algorithm for digital IIR filter design. Optik (stuttg) 126:2979–2984. https://doi.org/10.1016/j.ijleo.2015.07.022

Shehab M, Abu-Hashem MA, Shambour MKY, Alsalibi AI, Alomari OA, Gupta JND, Alsoud AR, Abuhaija B, Abualigah L (2023) A comprehensive review of bat inspired algorithm: variants, applications, and hybridization. Arch Comput Methods Eng 30:765–797. https://doi.org/10.1007/s11831-022-09817-5

Shishavan ST, Gharehchopogh FS (2022) An improved cuckoo search optimization algorithm with genetic algorithm for community detection in complex networks. Multimed Tools Appl 81:25205–25231. https://doi.org/10.1007/s11042-022-12409-x

Singh S, Ashok A, Kumar M, Rawat TK (2019) Adaptive infinite impulse response system identification using teacher learner based optimization algorithm. Appl Intell 49:1785–1802. https://doi.org/10.1007/s10489-018-1354-4

Snášel V, Rizk-Allah RM, Izci D, Ekinci S (2023) Weighted mean of vectors optimization algorithm and its application in designing the power system stabilizer. Appl Soft Comput 136:110085. https://doi.org/10.1016/j.asoc.2023.110085

Sudarsana Reddy K, Mahalakshmi R (2022) A MIMO-based compatible fuzzy logic controller for DFIG-based wind turbine generator. Presented at the (2022)

Tizhoosh HR (2005) Opposition-based learning: a new scheme for machine intelligence. In: international conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06). pp 695–701. IEEE (2005)

Torczon V (1997) On the convergence of pattern search algorithms. SIAM J Optim 7:1–25. https://doi.org/10.1137/S1052623493250780

Trojovský P, Dehghani M (2022) Pelican optimization algorithm: a novel nature-inspired algorithm for engineering applications. Sensors 22:855. https://doi.org/10.3390/s22030855

Turkoglu B, Kaya E (2020) Training multi-layer perceptron with artificial algae algorithm. Eng Sci Technol Int J. https://doi.org/10.1016/j.jestch.2020.07.001

Upadhyay P, Kar R, Mandal D, Ghoshal SP (2016) A new design method based on firefly algorithm for IIR system identification problem. J King Saud Univ Eng Sci 28:174–198. https://doi.org/10.1016/j.jksues.2014.03.001

Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1:67–82. https://doi.org/10.1109/4235.585893

Xu B, Heidari AA, Kuang F, Zhang S, Chen H, Cai Z (2022) Quantum Nelder-Mead Hunger Games Search for optimizing photovoltaic solar cells. Int J Energy Res 46:12417–12466. https://doi.org/10.1002/er.8011

Yang Y, Yang B, Niu M (2018) Adaptive infinite impulse response system identification using opposition based hybrid coral reefs optimization algorithm. Appl Intell 48:1689–1706. https://doi.org/10.1007/s10489-017-1034-9

Yang Y, Chen H, Heidari AA, Gandomi AH (2021) Hunger games search: visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. Expert Syst Appl 177:114864. https://doi.org/10.1016/j.eswa.2021.114864

Zaman HRR, Gharehchopogh FS (2022) An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems. Eng Comput 38:2797–2831. https://doi.org/10.1007/s00366-021-01431-6

Zhao R, Luo Q, Zhou Y (2017) Elite opposition-based social spider optimization algorithm for global function optimization. Algorithms 10:9. https://doi.org/10.3390/a10010009

Zhao R, Wang Y, Liu C, Hu P, Jelodar H, Yuan C, Li Y, Masood I, Rabbani M, Li H, Li B (2020) Selfish herd optimization algorithm based on chaotic strategy for adaptive IIR system identification problem. Soft Comput 24:7637–7684. https://doi.org/10.1007/s00500-019-04390-9

Zheng R, Jia H, Abualigah L, Liu Q, Wang S (2021) Deep ensemble of slime mold algorithm and arithmetic optimization algorithm for global optimization. Processes 9:1774. https://doi.org/10.3390/pr9101774

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.