INTERNATIONAL SCIENTIFIC AND VOCATIONAL JOURNAL (ISVOS JOURNAL)

Vol.: 9 Issue: 1 Date: 30.06.2025 Received: 02.11.2024 Accepted: 08.01.2025 Final Version: 09.01.2025

ISVOS Journal, 2025, 9(1): 12-23 – DOI: 10.47897/bilmes.1578027

Optimizing PI Controller for Stability and Overshoot in Step Response Using GA and PSO Techniques, A Comparative Study

Bilal Şenol [a,1], Uğur Demiroğlu [b]

[a] Software Engineering, Aksaray University, Aksaray, Türkiye
ORCID ID: 0000-0002-3734-8807
[b] Software Engineering, Kahramanmaraş İstiklal University, Kahramanmaraş, Türkiye
ORCID ID: 0000-0002-0000-8411

Abstract

In this paper, we present a comprehensive and in-depth investigation on the optimization of Proportional-Integral (PI) controller tuning for achieving stability and desired overshoot in the step response. The main objective of this study is to compare the effectiveness of Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) techniques in finding the optimal parameters for the PI controller. The PI controller is a widely used control algorithm that plays a crucial role in many industrial processes. Its tuning greatly affects the system's performance, particularly in terms of stability and overshoot. Therefore, finding the optimal tuning parameters is of utmost importance. To address this optimization problem, we propose the utilization of two popular metaheuristic algorithms, GA and PSO. These algorithms are known for their ability to efficiently search through large solution spaces and find near-optimal solutions. By applying these algorithms to the PI controller tuning problem, we aim to determine which technique yields better results in terms of stability and overshoot tuning. In our comparative study, we provide a detailed explanation of both GA and PSO algorithms, focusing on their working principles and mathematical formulations. We also describe how these algorithms can be applied to the PI controller tuning problem. Furthermore, we highlight the key differences between GA and PSO, shedding light on their strengths and weaknesses. To assess the performance of GA and PSO, we conduct several experiments using different benchmark functions and step response models. We measure the stability and overshoot metrics for various parameter settings obtained through GA and PSO. By thoroughly analyzing the obtained results, we draw meaningful conclusions regarding the effectiveness of each technique. Our findings demonstrate that both GA and PSO exhibit promising results in optimizing PI controller tuning. These observations provide valuable insights and guidelines for choosing the appropriate algorithm based on specific control requirements. In conclusion, this comparative study is thought to contribute to the field of control systems engineering by offering a comprehensive analysis of GA and PSO techniques in the context of PI controller tuning. By highlighting their strengths and weaknesses, it is aimed to provide researchers and practitioners with valuable information for making informed decisions when optimizing control parameters for stability and overshoot reduction purposes.

Keywords: "PI Controller, overshoot, step response, stability, genetic algorithm, particle swarm optimization."

1. Introduction

The study and examination of electronic systems, also known as system analysis, plays a vital role in various industries. One of the essential aspects of this process is understanding the step response. The step response is a crucial parameter that provides valuable insights into the behavior and performance of a system. In system analysis, the step response refers to the output of a system when a step input is applied to it. A step input is an abrupt change in the input signal, typically from zero to a constant value. By studying the step response, engineers and analysts can gain valuable information concerning the system's stability, transient response, and overall performance [1-3].

System stability plays a vital role in the efficient functioning of any given system. It is of utmost significance to ensure that a system remains stable and performs optimally over time. When a system lacks stability, it becomes susceptible to various issues, such as crashes, errors, and malfunctions. These issues can lead to significant disruptions in productivity, data loss, and even financial losses for businesses. One of the main reasons why system stability is crucial is its impact on user experience. A stable system provides a smooth and reliable platform for users to perform their tasks without interruption. Whether it's using software applications, accessing online services, or managing complex operations, a stable system ensures that users can complete their activities efficiently and effectively [4-6].

[1] Corresponding Author
E-mail Address: bilal.senol@aksaray.edu.tr

The step response exhibits several significant characteristics that make it an invaluable tool in system analysis. One of these characteristics is the overshoot [7]. Overshoot refers to the temporary excursion of the system output beyond the desired value before reaching stability. Understanding the magnitude and duration of overshoot is crucial in applications where precise control and stability are essential.

In control systems, the Proportional-Integral (PI) Controller holds great significance as it plays a crucial role in achieving precise control. This article aims to provide an exhaustive and comprehensive analysis of this controller, elucidating its functioning and impact on control systems. The Proportional-Integral (PI) Controller combines the proportional and integral control concepts to ensure improved control performance. By utilizing the proportional control aspect, the controller can respond to deviations in a system's output proportional to the error magnitude. This proportional action aids in minimizing steady-state errors and reducing overshoots. Apart from the proportional action, the PI controller incorporates the integral action to address accumulation of error over time, also known as integral windup. The integral control component continuously integrates the error signal, allowing the PI controller to effectively handle system disturbances and eliminate long-term steady-state errors [8-11].

The importance of metaheuristic methods in control theory lies in their ability to revolutionize the field of control systems. With their innovative approaches and powerful algorithms, these methods have proven to be invaluable in solving complex control problems that traditional methods fail to address. By harnessing the principles of optimization, search, and exploration, metaheuristic methods open up a whole new realm of possibilities for control theorists and practitioners. Whether it is enhancing the performance of autonomous systems, optimizing resource allocation, or designing robust controllers, metaheuristic methods provide a powerful toolkit for tackling real-world control challenges. From genetic algorithms to particle swarm optimization, these methods offer a diverse set of techniques that can be tailored to meet the specific needs of different control applications. With their ability to handle non-linear, multi-objective, and dynamic control problems, metaheuristic methods have become a cornerstone of modern control theory. As the complexity of control systems continues to grow, the importance of metaheuristic methods will only increase, helping us push the boundaries of what is possible in the world of control theory [12-15].

The Genetic Algorithm in controller tuning is a powerful and versatile method used to optimize the parameters of a controller [16]. By mimicking the process of natural selection and evolution, the Genetic Algorithm searches through a vast solution space to find the optimal set of controller parameters that maximizes the performance of the system. This algorithm works by iteratively generating a population of candidate solutions and evaluating their fitness based on a specified objective function. The fittest individuals are then selected to produce offspring, which inherit characteristics from their parents through crossover and mutation operations. This process continues for multiple generations until a satisfactory solution is found [17]. The Genetic Algorithm's ability to explore multiple possibilities simultaneously and find near-optimal solutions makes it an essential tool in controller tuning for complex and nonlinear systems [18, 19].

The Particle Swarm Optimization (PSO) algorithm is a popular technique used for controller tuning in various fields [20]. PSO is inspired by the natural behavior of bird flocking or fish schooling, where individuals coordinate their movements based on the collective knowledge of the group. In controller tuning, PSO is utilized to find the optimal parameters for a given control system. By using an iterative process, the algorithm explores the search space and adjusts the controller parameters to minimize a specified performance metric, such as error or overshoot. This iterative process continues until the algorithm converges to a set of parameters that yield the best control performance [21, 22].

This paper implements the advantages of the above mentioned optimization algorithms to shape the step response of a time delay plant controlled by a PI controller. The aim in shaping is to satisfy the overshoot of the step response towards researchers' desire as well as to achieve stability. The literature surrounding step response shaping using PI controllers is rich with insights into the evolution, design, and optimization of PID control systems. The study in [23] discusses the optimization of Proportional-Integral (PI) and Proportional-Integral-Derivative (PID) controllers to satisfy certain transient performance standards. This work presents tuning approaches that, akin to the Ziegler–Nichols technique, need just parameters derived from the plant's step response. The authors enhance the methodology to include plants with underdamped step responses and present a systematic way for adjusting the controller gain to meet specified transient performance criteria, including less overshoot and enhanced settling time. The foundational work in [24] highlights the integral role of the PID controller in industrial applications, emphasizing its ability to stabilize system outputs through manipulation of the error signal. Hill articulates the significance of the proportional, integral, and derivative gains, with particular emphasis on the integral term, which is crucial for driving the error signal to zero—a key aspect for achieving accurate step response shaping. Furthermore, Hill notes that while the derivative term can enhance transient response shaping, its tendency to amplify high-frequency noise poses challenges in practical applications. This early exploration sets the stage for understanding the complexities involved in tuning PID controllers effectively. In the valuable study in [25], an analytical approach for optimizing Proportional-Integral (PI) controllers in the Field Oriented Control (FOC) of Permanent Magnet Synchronous Motors (PMSMs) is presented. The authors analytically define the attributes of the closed-loop response to ascertain PI controller gains that satisfy specified dynamic criteria, including maximum overshoot and rising time. Due of the nonlinear characteristics of the resultant equations, the Newton-Raphson numerical approach is utilized for their resolution. The suggested method guarantees that, through the use of decoupled models of the PMSM, the closed-loop system operates linearly, ensuring stability throughout an extensive operational range. Building upon this foundation, [26] addresses the critical challenge of controller tuning in process control systems. He underscores the importance of optimizing PID controller parameters to enhance system performance, particularly in the face of inherent difficulties such as system nonlinearity and

unstable open-loop systems. Practical methods for near-optimal tuning of Proportional-Integral (PI) and Proportional-Integral-Derivative (PID) controllers were studied in [27]. The authors present simple methods for adjusting PI and PID controllers to attain a balance between output performance and control activity, hence providing resilience in the mid-frequency range. For systems with all poles located on the negative real axis, a singular step response is enough for plant characterization. In systems exhibiting integral action, doing an impulse response or relay experiment is advisable. [28] introduces an improved gravitational search algorithm aimed at automating the tuning process, thereby providing optimized parameters for settling time, percentage overshoot, and steady-state error. This contribution not only complements [25] 's findings but also reflects the ongoing need for innovative solutions to improve the performance of widely used PID controllers, particularly given that a significant proportion remain poorly tuned in practice. Further advancing the discourse, [28] provides a contemporary perspective on the design of classical controllers, noting that over 95% of control loops in process control are of the PID type, with a predominant focus on PI control. He elaborates on the operational principles of the proportional, integral, and derivative actions, detailing how each component contributes to overall system stability and error correction. This article emphasizes the straightforward structure and ease of implementation of PI controllers, reinforcing their popularity in industrial settings. [28] also highlights the ongoing development of new tuning methods, which are critical for enhancing the design and effectiveness of these classical control systems.

The organization of this paper is as follows. Second chapter introduces the PI controller and briefly reminds the step response. The third chapter briefly recalls the GA and PSO methods. A case study is given in the fourth chapter and the fifth chaptr includes the conclusions.

2. PI Controller and The Step Response

The literature on the structure and design of Proportional-Integral (PI) controllers reveals a rich history of development and refinement, particularly in the context of industrial applications. The work in [29] establishes a foundational understanding of fixed structure controllers, specifically focusing on the integration of a PI velocity feedback controller with a PI outer tension loop to effectively regulate web tension and velocity. Their investigation into systematic design methods highlights the importance of satisfying pre-specified performance criteria through parametric approaches. This paper underscores the versatility of these design techniques, suggesting their applicability extends beyond tension control to various control loops, thereby enriching the discourse on fixed structure controller design that has evolved since the 1960s. Building on this foundation, [24] shifts the focus to the broader category of PID controllers, which incorporate proportional, integral, and derivative elements. The author articulates the significance of the integral term within the PID framework, emphasizing its role in driving the error signal to zero and thus ensuring system stability. Hill's analysis traces the historical development of PID controllers, from their origins in mechanical devices to their modern digital implementations, illustrating the evolution of tuning methods that have accompanied this progression. The simplicity of the PID structure, which requires only three gain parameters, is highlighted as a key advantage, making it a popular choice in various industrial applications. This exploration into real-time optimal auto-tuning further complements the understanding of controller structures, emphasizing the need for efficient manipulation of error signals to stabilize system outputs.

The transfer function of the PI controller is given as follows.

$$C(s) = k_p + \frac{k_i}{s} \tag{1}$$

As known, k_p is the proportioal coefficient and k_i is the integral coefficient. The plant used in this study is a classical first order plus time delay plant which has the following structure.

$$P(s) = \frac{K}{Ts+1} e^{-Ls} \tag{2}$$

Here, K is the gain constant, T is the time constant and L is the time delay. After this brief introduction, an explanation about the step response should be benefical. First-order plus time delay (FOPTD) systems are among the most important and widely used dynamic processes in various major industries, such as power systems, automotive and aerospace systems, biological systems, and chemical processes, as well as refinery industries, to model their dynamic processing. Different technological innovations and rapid advances in computational power lead to faster, smarter, and more innovative control systems that require less energy, time, and effort to improve the dynamic quality of processing systems. Due to this ever-evolving technology, control systems are now mandatory for the vast majority of process industries. A control system is an essential part of an industry used to manage, automate, and regulate workflow. If any dynamic processing system is not under control, it is not possible to obtain accurate, steady-state, and reduced-error controlled outputs. During a control design, the operating plant is first investigated to find its dynamic characteristics. After determining its dynamic characteristics, the details of the operating plant are considered for selecting a suitable controller. During the investigation of a plant, it is often represented by a first or second-order dynamic

response plus a time delay. The dynamic system defined by the first-order plus time delay (FOPTD) system is an essential part of modern control theory. Many controllers have been designed for this type of dynamic system, which is quickly and efficiently applied in many engineering control and automation applications. The step changes and transfer functions of FOPTD systems are used in numerous fields, such as medicine, chemical processes, refinery batch process productions, servo mechanisms, machine tools, biosystems, agriculture, aircraft, marine, and automotive industries. In many engineering control and automation applications, the input-output time response and transfer function equation of a first-order plus time delay (FOPTD) system are important and necessary. Even so, no single tool or method is used effectively to achieve the major control system objectives simultaneously for this practical class of dynamic control systems [30-33].

The literature review provides a detailed examination of step response analysis within the framework of control systems, particularly emphasizing the role of classical feedback control. A critical element of the analysis is the use of the frequency response function (FRF) to represent the system's behavior accurately. This approach allows for a more realistic assessment of system performance under operational conditions, moving beyond reliance on estimated transfer function models. The findings indicate that proper tuning of the PID controller can ensure that the system meets essential specifications related to gain and phase margins, peak sensitivity, and overall stability. The authors also highlight the classical PID controller's versatility across various industrial applications, demonstrating its effectiveness in improving both steady-state error and transient response through the integration of PI and PD components. In conclusion, the literature emphasizes the importance of step response analysis in control systems, particularly in relation to the performance of classical feedback controllers like the PID controller [34]. A sample illustration of the step response is given in Fig. 1 [35].

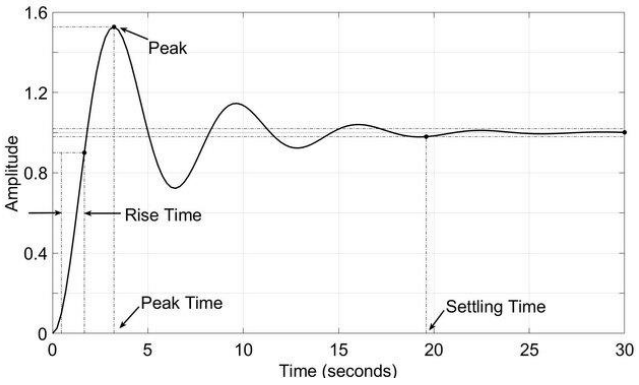

Fig. 1. The step response

This paper deals with the point named as “Peak” in the figure. The aim is to tune the overshoot (peak) value towards our desire using the well known optimization algorithms. Next section gives information about the genetic algorithm and particle swarm optimization.

3. Genetic Algortihm and Particle Swarm Optimization

3.1. The Genetic Algorithm

Originally proposed in the early 1960s, genetic algorithms (GAs) are a popular evolutionary algorithm designed to solve large, complex search and optimization problems that are difficult to model and solve propositionally using conventional methodology. Inspired by the Darwinian process of natural selection, GAs are based on the approach used by nature for the evolution of organisms on Earth. Genetic algorithms are based on these phenomena observed in the natural world, and these concepts were employed to solve technological problems. A search in GAs evolves a population of strings, under the process of natural selection and genetics, to optimize or solve a certain problem that requires optimization or search. One important aspect of using genetic algorithms is that no domain knowledge is assumed to solve the problem at hand, and only a set of primitive operations is applied to evolve a population of strings until a criterion is satisfied. The underlying principle of GAs is simple: the most fit will survive and propagate to the next generation, and it is believed that the long-term survival of a population in a complex environment will result in the evolution of a population. A diverse range of machine learning applications often makes use of genetic algorithms at some stage in their process. This adaptive process has been found to be successful, often when humans were unable to develop optimal methodologies due to lack of inherent knowledge or when most of the problem was uncertain and subjective. Key features of GAs are listed below.

- Evolution: The search technique is inspired by the natural selection process.
- Parallel search: The population evolves to represent the potential search space.

- Multi-objective technique: More than one solution is simultaneously examined, providing a diverse range of solutions.
- Complex search space: Suitable for the irregular search space, where the derivative operation design is not feasible.
- Domain knowledge: No need for domain knowledge, apart from basic operations that can be performed.
- Some of the reasons to use GAs as opposed to traditional deterministic or stochastic optimization algorithms are:
- In a wide population, various loners can be obtained and propagated for further examination.
- More than one global solution at a time is found using a diverse population.
- Efficient in a practical real-world space, where the function might not have a continuous derivative and is not dominated by the optimal local solution.
- Optimization or learning does not require any additional conditions that have to be fulfilled, such as knowledge that will precede the final function or local minima.
- Some of the basic terminology used within GA description:
- Gene: Forms the basic building block of a chromosome; they are the basic underlying data that represent the solution.
- Chromosome: A chromosome is considered a finite sequence of genes in a fixed order.
- Population: A population is a group of K candidate solutions considered for the solution. Each individual can be considered as a chromosome that represents a possible potential solution.
- Allele: It represents a value of a gene.

The pseudo code of the GA can be written in the following way.

1. Initialize the population with random solutions.
2. Evaluate the fitness of each individual in the population.
3. Repeat until the stopping condition is met:
    a. Select individuals from the population based on their fitness (selection).
    b. Apply crossover to selected individuals to produce offspring.
    c. Apply mutation to offspring to introduce variability.
    d. Evaluate the fitness of the new individuals.
    e. Replace some or all of the population with the new individuals.
4. Return the best individual as the solution.

DETAILS OF STEPS:

- INITIALIZE_POPULATION(size, bounds):

Create `size` individuals with random values within `bounds`.

$$\mathrm{X} = \left[x_1, x_2, x_3, ...x_n \right] \tag{3}$$

- FITNESS_FUNCTION(individual):

Compute the fitness of `individual`.

$$f(x) \to \square \tag{4}$$

- SELECTION(population, fitness_values, method):

Select individuals using a method like tournament, roulette wheel, or rank-based selection.

$$P(\mathrm{X}_i) = \frac{f(\mathrm{X}_i)}{\sum_{j=1}^{N} f(\mathrm{X}_j)} \tag{5}$$

- CROSSOVER(parent1, parent2, method):

Combine `parent1` and `parent2` using methods like single-point, two-point, or uniform crossover.

$$x_i^{offspring} = \begin{cases} x_i^{(p1)} & if\ rand() < 0.5 \\ x_i^{(p2)} & otherwise \end{cases} \tag{6}$$

- MUTATION(individual, mutation_rate):

Randomly alter genes of `individual` with a probability `mutation_rate`.

$$x_i = \begin{cases} 1 - x_i & if\ rand() < P_{mut} \\ x_i & otherwise \end{cases} \tag{7}$$

- REPLACEMENT(population, offspring, strategy):

Replace individuals in the population using strategies like elitism, generational replacement, or steady-state replacement.

Additionally, some terminologies need to be understood when using GAs for further explanations. These terminologies are chromosome, gene, alleles, genotype, phenotype, and other GA terms. These definitions serve as a fundamental base for developing a sound understanding of genetic algorithms [19, 36, 37].

3.2. The Particle Swarm Optimization

In recent years, the Particle Swarm Optimization (PSO) algorithm has gained significant attention from researchers due to its ability to solve complex optimization problems. PSO is inspired by the social behavior of birds, which can be collectively seen as a flock. The flock consists of birds moving together in a cohesive manner, and the birds collectively decide the direction of flight. The collective movements of a bird swarm may give the impression that collective intelligence is at work. Expanding on this phenomenon of birds flocking to a closed space, computer scientists have developed a collective intelligence-based search algorithm. They have been successful in the planning and execution of such algorithms for optimization problems. PSO attempts to optimize problems by simulating the social behavior of birds [38-40].

With the advancement of sciences like engineering, biology, and others in the last few decades, the problems formulated in these fields are complex in terms of the size of the search space. Traditional optimization methods face a number of challenges while solving such problems. Therefore, researchers are looking to overcome these challenges by utilizing some nature-inspired algorithms to aid in the solution of such problems. In the past, many meta-heuristic algorithms have been proposed by researchers. The motivation for preparing a comprehensive study of PSO is to provide insight into various computational intelligence techniques. The specific objectives of this comprehensive study are to highlight the differences existing among various relatively recent issues and to signal the possible future trends for research scholars. It is a powerful algorithm and may provide optimal solutions for complex problems where different soft computing techniques or some traditional methods may yield inferior solutions.

The basic terms used in PSO include particle, position, velocity, pBest, gBest, c1, c2, w, fitness function, domain space, and swarm. Many researchers in the field of PSO consider these terms to be the most important terms that are used in PSO. These terms are defined as follows:

- Particle: a single member of the swarm.
- Position: the location of a particle in the search space.
- Velocity: a particle's position and how it is modified during a search. Formally, it is the direction of future estimated improvements.
- pBest: personal best solution that a particle has encountered during the search process.
- gBest: global best solution in a swarm or used for a local fitness problem. A global best solution is a solution that is found from all particles in a swarm.
- c1 and c2: learning factors that control the contributions of pBest and gBest. The two learning factors c1 and c2, both in the range of [0, 2], control the cognitive and social components of PSO. The cognitive component promotes the exploitation process, and the value of a given pBest for each individual exploration is reflected by the value of c1 and also by (Xi (k) - pBesti (k)). The social component represents the exploration process, and the value of gBest of the group solutions exploration is reflected by the value of c2 and also by (Xi (k) - gBest (k)).
- w: inertia weight factor that adjusts the PSO performance. Many researchers found that the values of the inertia weight factor that are in the range of [0.4, 0.9] can obtain a higher search performance in the search process. The inertia weight factor reflects on the ability of particles in the search process through its linear or nonlinear decrease in velocity over time.
- Fitness Function: a fitness function assigned to each particle which indicates how well a possible solution fits the desired form.
- Domain Space: a space that includes all possible choices of a problem parameter.
- Swarm: a group of particles in the search space. Each particle in the swarm is characterized by its position, velocity, and its best position pBest that it has found during the generations of the search process.

All PSO parameters have a direct tuning effect on the search process performance. The important parameters of PSO based on the characteristics of the search space include the swarm size and the velocity, which is affected by three parameters: acceleration constant, pBest, and gBest. The behavior of each particle in the swarm is controlled by the cognitive (pBest) and social (gBest) acceleration coefficients, which are the degree of importance of the cognitive and social speed [41-43].

The pseudo code of the PSO can be written in the following way.

1. Initialize the swarm:
    a. Set the number of particles, positions, velocities, and other parameters (e.g., inertia weight, cognitive and social coefficients).
    b. Randomly initialize each particle's position and velocity within the search space.
    c. Evaluate the fitness of each particle's position.
    d. Set each particle's initial personal best position (pBest) to its current position.
    e. Identify the global best position (gBest) among all particles.
2. Repeat until the stopping condition is met:
    a. For each particle:
        i. Update the particle's velocity:
        v[i] = w * v[i]
        + c1 * r1 * (pBest[i] - x[i])
        + c2 * r2 * (gBest - x[i])
        ii. Update the particle's position:
        x[i] = x[i] + v[i]
        iii. Evaluate the fitness of the new position.
        iv. Update the particle's personal best (pBest) if the new position is better.
    b. Update the global best position (gBest) if any particle's new position is better than the current gBest.
3. Return the global best position (gBest) as the solution.

Key Formulations in the Pseudocode:

1. Velocity Update Rule

For particle i, the velocity update formula is:

$$v_i(t+1) = w \cdot v_i(t) + c_1 \cdot r_1 \cdot (pBest - x_i(t)) + c_2 \cdot r_2 \cdot (gBest - x_i(t)) \tag{8}$$

where, w is the inertia weight (controls the influence of the previous velocity). c_1, c_2 are the cognitive and social coefficients (control the influence of $pBest$ and $gBest$). r_1, r_2 are random numbers uniformly distributed in [0, 1].

2. Position Update Rule

The position of particle i is updated as:

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{9}$$

3. Fitness Evaluation

The fitness of each particle is evaluated using a predefined fitness function, $f(x)$

4. Personal Best

Each particle maintains its best position found so far:

$$pBest_i = \arg\min/\max f(x_i) \rightarrow (\text{depending on the problem}) \tag{10}$$

5. Global Best

$$gBest_i = \arg\min/\max f(pBest_i) \qquad for\ all\ i \tag{11}$$

4. Case Study

Example 1: This example implements the GA and PSO to tune the PI controller for the below FOPTD plant.

$$G^1(s) = \frac{0.99932}{1.0842s+1} e^{-0.1922} \tag{12}$$

The step response will be plotted for 25 seconds and the peak of the step response is set to be 1.2 in magnitude. The GA resulted with the following PI controller at the end of the optimization process.

$$C^1_{GA\_1}(s) = 0.534251 + \frac{1.599687}{s} \tag{13}$$

Simiarly, the result of the PSO after 1000 iterations is,

$$C^1_{PSO\_1}(s) = 3.6778 + \frac{3.9271}{s} \tag{14}$$

Fig. 2 shows the step response obtained by using $C_{GA}(s)$ and Fig. 3 shows the one controlled with $C_{PSO}(s)$.

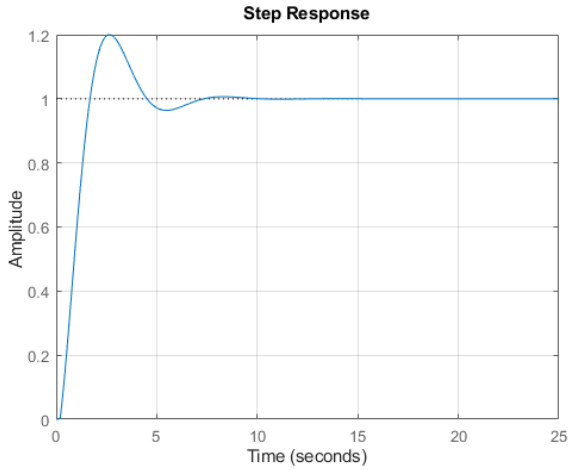

Fig. 2. The step response obtained with genetic algorithm for 20% overshoot

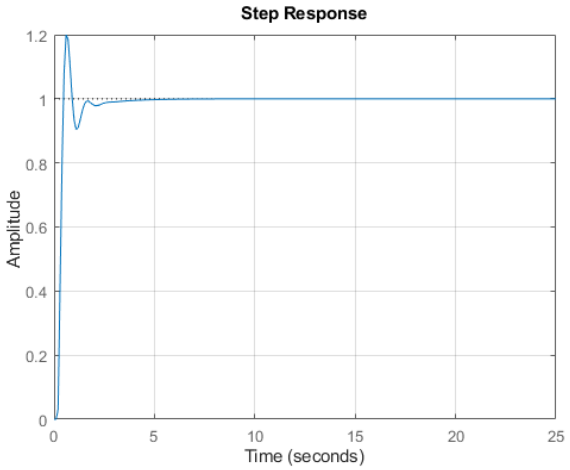

Fig. 3. The step response obtained with particle swarm optimization for 20% overshoot

We can see from the two step responses that both systems show stable response and their overshoot is exactly tuned to be 1.2 as desired. For comparison, it can be said that the genetic algorithm resulted with a longer rise time and settling time. Let us now tune the controller to obtain the step response with no overshoot. The PI controller obtained using the genetic algorithm is given as follows.

$$C^1_{GA\_2}(s) = 2.006661 + \frac{1.850810}{s} \tag{15}$$

Simiarly, the PI controller obtained using the particle swarm optimization is,

$$C^1_{PSO\_2}(s) = 1.847255 + \frac{1.703791}{s} \tag{16}$$

This time, the step response of the system obtained with the genetic algorithm is drawn in Fig. 4. Similarly, the step response obtained by tuning the controller with the PSO is given in Fig. 5. The two methods successfully achieved the system to have a step response with no overshoot. It can also be seen that the two step responses approximately match eachother.

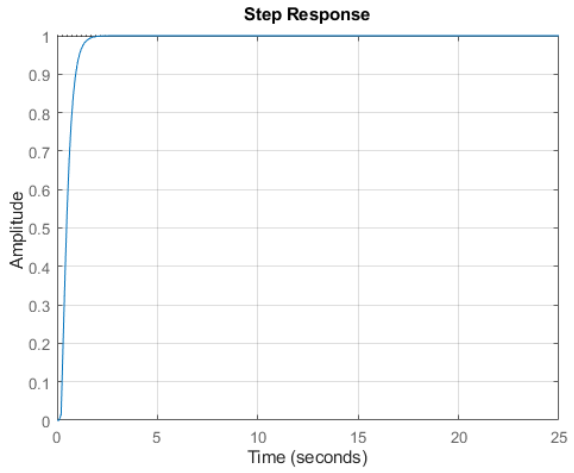

Fig. 4. The step response obtained with genetic algorithm for no overshoot

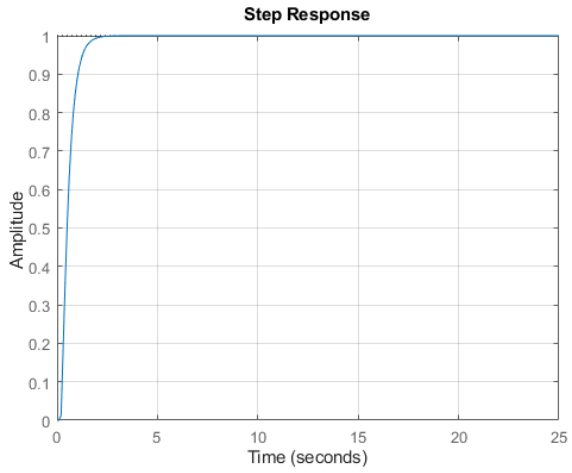

Fig. 5. The step response obtained with particle swarm optimization for no overshoot

It would be useful to apply the methods on anothr example.

Example 2: Consider the following time delay plant [44].

$$G^2(s) = \frac{1}{s+1} e^{-0.1} \tag{17}$$

The overshoot for this example is desired to be 10%. For this aim, following controllers have been obtained.

$$C^2_{GA}(s) = 0.522484 + \frac{1.334183}{s} \tag{18}$$

Simiarly, the result of the PSO after 1000 iterations is,

$$C^2_{PSO}(s) = 6.935578 + \frac{3.837198}{s} \tag{19}$$

Using the controllers in Eq. 18 and Eq. 19, following step responses are found.

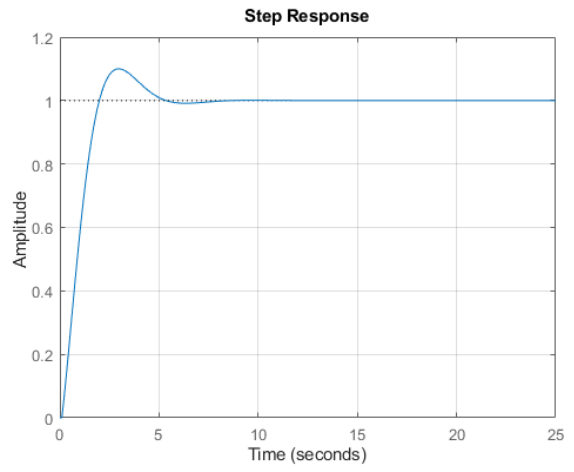

Fig. 6. The step response obtained with genetic algorithm for 10% overshoot

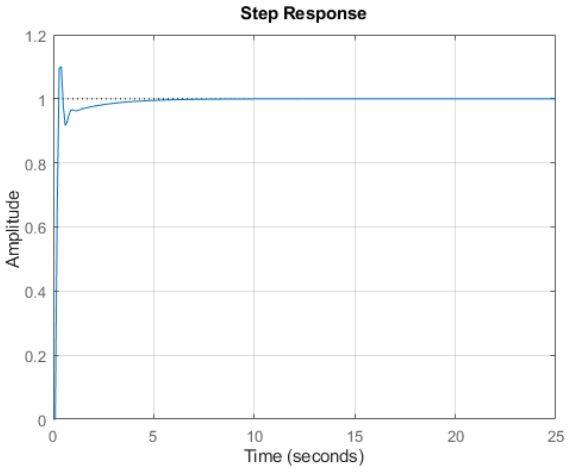

Fig. 7. The step response obtained with particle swarm optimization for 10% overshoot

It can be clearly seen in Fig. 6 and Fig. 7 that the overshoot of the step response is successfully limited to be 10%. Comparing two figures show that the step response obtained with the genetic algorithm has a smoother increase but relatively a larger steady

state time. In contrast, the step response obtained with the particle swarm optimization has a sharp increase and a smaller steady state time. Thus, the choice can be made by considering the needs of the system.

According to the report, the findings of the comparison study between the Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) for the tuning of the PI controller are quite illuminating. The overshoot levels that were intended (for example, 20%, no overshoot, 10%) were successfully obtained by both strategies, and the step response remained stable throughout the accomplishment. Nevertheless, discrepancies of a significant nature were identified in their performance. The use of GA often led to longer rising and settling durations, which indicated smoother transitions but slower reactions. PSO, on the other hand, displayed more abrupt climbs and shorter settling durations, which might be useful in systems that require speedier stabilization. These findings highlight how important it is to pick the optimization strategy based on the individual requirements of the system, striking a balance between smoothness and speed. It is possible that future study may concentrate on adapting these algorithms to a wide variety of step response qualities in order to facilitate the expansion of their practical applications.

5. Conclusions

This paper provides a thorough examination of the methodologies for optimizing Proportional-Integral (PI) controller tuning to achieve stability and desired overshoot in the step response. The primary objective of the study is to compare and contrast the effectiveness of two heuristic optimization techniques, namely Genetic Algorithm (GA) and Particle Swarm Optimization (PSO), in finding the best parameters for the PI controller, which plays a critical role in industrial processes. The tuning process significantly impacts the stability and overshoot of the system, emphasizing the need for identifying the most suitable parameters. To tackle this challenge, the paper proposes the utilization of GA and PSO, both of which are renowned for their capability to efficiently explore large solution spaces. The study provides an in-depth explanation of the underlying principles of both algorithms and elucidates how they can be applied to solve the PI controller tuning problem. Moreover, it sheds light on the distinctions between GA and PSO, meticulously examining their respective strengths and weaknesses. In order to evaluate the performance of these optimization techniques in terms of stability and overshoot metrics, a series of experiments are conducted, employing various models and functions. The results obtained from these experiments demonstrate promising outcomes for both GA and PSO in the optimization of PI controller tuning, thereby offering valuable insights to researchers and practitioners alike, enabling them to make informed decisions based on the specific control requirements they may encounter. Ultimately, this comparative study provides a comprehensive analysis and delivers a wealth of valuable information to individuals involved in the field of control systems engineering.

The study lays the groundwork for a discussion of the findings by contrasting the effectiveness of Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) in the process of tuning PI controllers for step response shaping. Nevertheless, a comparison with the current literature that is more specific might increase the depth of the conversation as well as the contextual relevance of the topic. According to the examination of the relevant literature, works such as [23] place an emphasis on transient performance requirements in PI and PID tuning. These studies also propose analytical approaches to achieve optimal overshoot and settling time respectively. In a similar vein, [28] outlines developments in traditional PID designs, which are the most common kind used in industrial settings. In comparison to the current investigation, the metaheuristic algorithms (GA and PSO) provide an automated and adaptable method that can accommodate different system dynamics without the requirement of pre-specified analytical parameters. The article effectively illustrates that both GA and PSO are capable of successfully achieving desired overshoot levels (for example, 20%, no overshoot, 10%). This is demonstrated in the conclusions section of the paper. Nevertheless, there are obvious trade-offs: GA exhibits smoother responses with longer rising and settling periods, whereas PSO gives faster answers but with sharper rises. Both methodologies have their advantages and disadvantages. The findings are consistent with the observations made in the literature, which state that GA frequently outperforms in exploration, whereas PSO strikes a balance between exploration and exploitation in order to achieve faster convergence [16, 20]. The publication might compare these results with analytical methods [23, 25] or other metaheuristics [26], describing circumstances in which GA or PSO might perform better than traditional approaches. This would enrich the presentation of the results. In order to emphasize the practical benefits of these algorithms, it would be beneficial to highlight their adaptability and flexibility in dealing with nonlinearities and requirements that involve several objectives. An further contextualization of the paper's contributions would be achieved by the use of benchmarks or measures from previous research, which would also make the conclusions more robust and generalizable.

References

[1] G. Cui, J. Yu and Q. G. Wang, “Finite-time adaptive fuzzy control for MIMO nonlinear systems with input saturation via improved command-filtered backstepping,” IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 2, pp. 980-989, 2020.

[2] Y. Ji, X. Jiang and L. Wan, “Hierarchical least squares parameter estimation algorithm for two-input Hammerstein finite impulse response systems, Journal of the Franklin Institute, vol. 357, no. 8, pp. 5019-5032, 2020.

[3] Y. Ji and Z. Kang, “Three-stage forgetting factor stochastic gradient parameter estimation methods for a class of nonlinear systems,” International Journal of Robust and Nonlinear Control, vol. 31, no. 3, pp. 971-987, 2021.

[4] B. Obrenovic, J. Du, D. Godinic, D. Tsoy, M. A. S. Khan, and I. Jakhongirov, “Sustaining enterprise operations and productivity during the COVID-19 pandemic:“Enterprise Effectiveness and Sustainability Model,” Sustainability, vol. 12, no. 15, p. 5981, 2020.

[5] M. Kamalahmadi, M. Shekarian, and M. Mellat Parast, “The impact of flexibility and redundancy on improving supply chain resilience to disruptions,” International Journal of Production Research, vol. 60, no. 6, pp. 1992-2020, 2022.

[6] Z. Yu, A. Razzaq, A. Rehman, A. Shah, K. Jameel and R. S. Mor, “Disruption in global supply chain and socio-economic shocks: a lesson from COVID-19 for sustainable production and consumption,” Operations Management Research, pp. 1-16, 2021.

[7] M. Kosmecki, R. Rink, A. Wakszyńska, R. Ciavarella, M. Di Somma, C. N. Papadimitriou, and G. Graditi, “A methodology for provision of frequency stability in operation planning of low inertia power systems,” Energies, vol. 14, no. 3, p. 737, 2021.

[8] M. Hu, W. Hua, G. Ma, S. Xu, and W. Zeng, “Improved current dynamics of proportional-integral-resonant controller for a dual three-phase FSPM machine,” IEEE Transactions on Industrial Electronics, vol. 68, no. 12, pp. 11719-11730, 2020.

[9] J. Park, H. Kim, K. Hwang, and S. Lim, “Deep reinforcement learning based dynamic proportional-integral (PI) gain auto-tuning method for a robot driver system,” IEEE Access, vol. 10, pp. 31043-31057, 2022.

[10] A. W. Khawaja, N. A. M. Kamari, M. A. A. M. Zainuri, S. Abd Halim, M. A. Zulkifley, S. Ansari, and A. S. Malik, “Angle stability improvement using optimised proportional integral derivative with filter controller,” Heliyon, 2024.

[11] S. M. Y. Younus, U. Kutbay, J. Rahebi, and F. Hardalaç, “Hybrid gray wolf optimization–proportional integral based speed controllers for brush-less dc motor,” Energies, vol. 16, no. 4, p. 1640, 2023.

[12] D. F. Zambrano-Gutierrez, J. M. Cruz-Duarte, J. G. Avina-Cervantes, J. C. Ortiz-Bayliss, J. J. Yanez-Borjas, and I. Amaya, “Automatic design of metaheuristics for practical engineering applications,” IEEE Access, vol. 11, pp. 7262-7276, 2023.

[13] E. Bogar and S. Beyhan, “Adolescent Identity Search Algorithm (AISA): A novel metaheuristic approach for solving optimization problems,” Applied Soft Computing, vol. 95, p. 106503, 2020.

[14] A. A. Ameen, “Metaheuristic Optimization Algorithms in Applied Science and Engineering Applications,” Doktoral Thesis, 2024.

[15] A. Mojtahedi, M. Dadashzadeh and M. Kouhi, “Developing a predictive method based on the vibration behavior of a naval ship hull model using hybrid fuzzy meta-heuristic algorithms,” Ocean Engineering, vol. 311, p. 118994, 2024.

[16] O. Rodríguez-Abreo, J. M. Garcia-Guendulain, "Genetic algorithm-based tuning of backstepping controller for a quadrotor-type unmanned aerial vehicle," Electronics, 2020. mdpi.com

[17] Y. Tian, Y. Zhang, Y. Su, and X. Zhang, "Balancing objective optimization and constraint satisfaction in constrained evolutionary multiobjective optimization," in IEEE Transactions on ..., 2021. researchgate.net

[18] G. D’Angelo, and F. Palmieri, “GGA: A modified genetic algorithm with gradient-based local search for solving constrained optimization problems,” Information Sciences, vol. 547, pp. 136-162, 2021.

[19] A. Sohail, “Genetic algorithms in the fields of artificial intelligence and data sciences,” Annals of Data Science, vol. 10, no. 4, pp. 1007-1018, 2023.

[20] A. G. Gad, "Particle swarm optimization algorithm and its applications: a systematic review," Archives of computational methods in engineering, 2022. springer.com

[21] H. Maghfiroh, M. Nizam, M. Anwar, and A. Ma’Arif, “Improved LQR control using PSO optimization and Kalman filter estimator,” IEEE Access, vol. 10, pp. 18330-18337, 2022.

[22] M. N. Muftah, A. A. M. Faudzi, S. Sahlan, and M. Shouran, “Modeling and fuzzy FOPID controller tuned by PSO for pneumatic positioning system,” Energies, vol. 15, no. 10, p. 3757, 2022.

[23] Basilio, J. C., and Matos, S. R. (2002). Design of PI and PID controllers with transient performance specification. *IEEE Transactions on education*, *45*(4), 364-370.

[24] A. Jamison Hill, “Towards Real Time Optimal Auto-tuning of PID Controllers,” Doctoral Thesis, 2013.

[25] Salimi, H., Zakipour, A., and Asadi, M. (2022). A novel analytical approach for time-response shaping of the pi controller in field oriented control of the permanent magnet synchronous motors. *Journal of Electrical and Computer Engineering Innovations (JECEI)*, *10*(2), 463-476.

[26] M. Saiful Islam Aziz, “Improved gravitational search algorithm for proportional integral derivative controller tuning in process control system”, Doctoral Thesis, 2016.

[27] Kristiansson, B., and Lennartson, B. (2006). Robust tuning of PI and PID controllers: using derivative action despite sensor noise. *IEEE Control Systems Magazine*, *26*(1), 55-69.

[28] I. De Jesus Diaz Rodriguez, “Modern Design of Classical Controllers,” 2017.

[29] R. Pagilla, P. Cimino and D. Knittel, ”Design of fixed structure controllers for web tension control,” 2007.

[30] Z. Chen, Y. S. Hao, Z. Su and L. Sun, “Data-driven iterative tuning based active disturbance rejection control for FOPTD model,” ISA transactions, vol. 128, pp. 593-605., 2022.

[31] H. Meneses, O. Arrieta, F. Padula, A. Visioli and R. Vilanova, “Fopi/fopid tuning rule based on a fractional order model for the process,” Fractal and Fractional, vol. 6, no. 9, p. 478, 2022.

[32] P. Ghorai, “Online Parameters Estimation of Time-Delayed Dynamics of Processes for Industrial Use, “ International Journal of Industrial Engineering, vol. 29, no. 4, 2022.

[33] T. George and V. Ganesan, “Optimal tuning of PID controller in time delay system: A review on various optimization techniques,” Chemical Product and Process Modeling, vol. 17, no. 1, pp. 1-28, 2022.

[34] L. Abdullah, Z. Jamaludin, T. H. Chiew, N. A. Rafan and M. Y. Yuhazri, “Extensive Tracking Performance Analysis of Classical feedback control for XY Stage ballscrew drive system,” Applied Mechanics and Materials, vol. 229, pp. 750-755, 2012.

[35] U. Agrawal, P. Etingov and R. Huang, “Advanced Performance Metrics and Sensitivity Analysis for Model Validation and Calibration,” Authorea Preprint, 2023.

[36] M. Gen and L. Lin, “Genetic algorithms and their applications,” In Springer handbook of engineering statistics (pp. 635-674). London: Springer London, 2023.

[37] Z. Z. Wang and A. Sobey, “A comparative review between Genetic Algorithm use in composite optimisation and the state-of-the-art in evolutionary computation,” Composite Structures, vol.233, 2020.

[38] H. R. R. Zaman and F. S. Gharehchopogh, F. S. (2022). An improved particle swarm optimization with backtracking search optimization algorithm for solving continuous optimization problems,” Engineering with Computers, vol. 38, no. 4, pp. 2797-2831 2022.

[39] Z. Yu, Z. Si, X. Li, D. Wang and H. Song, “A novel hybrid particle swarm optimization algorithm for path planning of UAVs,” IEEE Internet of Things Journal, vol. 9, no. 22, pp. 22547-22558, 2022

[40] A. G. Gad, “Particle swarm optimization algorithm and its applications: a systematic review,” Archives of computational methods in engineering, vol. 29, no. 5, pp. 2531-2561, 2022.

[41] H. Zhao, H. Yao, Y. Jiao, T. Lou, andY. Wang, “An Improved Beetle Antennae Search Algorithm Based on Inertia Weight and Attenuation Factor,” Mathematical Problems in Engineering, vol. 1, 2022.

[42] J. Chrouta, F. Farhani, F. and A. Zaafouri, “A modified multi swarm particle swarm optimization algorithm using an adaptive factor selection strategy,” Transactions of the Institute of Measurement and Control, vol. 0, no. 0, 2021.

[43] B. Durmuş, “Opposite Based Crow Search Algorithm for Solving Optimization Problem,” International Scientific and Vocational Studies Journal, vol. 5, no. 2, pp. 164-170, 2021.

[44] B. Şenol and U. Demiroğlu, “Analytical Design of PI Controllers for First Order plus Time Delay Systems,” International Scientific and Vocational Studies Journal, vol. 2, no. 2, pp. 40–47, 2018.