International Journal of Engineering and Technology Innovation, vol. 16, no. 1, 2026, pp. 113-133

Multi-Target Particle Swarm Optimization with Machine Learning Surrogates for Efficient Concrete Mix Design

Malik Mushthofa[1,2,*], John Thedy[1,4], Han Ay Lie[1], Purwanto[1], Marc Ottelé[3], Mochamad Teguh[2]

[1]Department of Civil Engineering, Diponegoro University, Semarang, Indonesia

[2]Faculty of Civil Engineering and Planning, Islamic University of Indonesia, Yogyakarta, Indonesia

[3]Faculty of Civil Engineering and Geosciences, Delft University of Technology, Delft, Netherlands

[4]Formerly Department of Bioenvironmental Systems Engineering, National Taiwan University, Taipei, Taiwan, ROC

Received 22 November 2025; received in revised form 05 January 2026; accepted 06 January 2026

DOI: https://doi.org/10.46604/ijeti.2026.15905

Abstract

This study presents a multi-target particle swarm optimization (MT-PSO) approach for efficient concrete mix design. It simultaneously designs mixes with multiple predefined strengths under a constant water-cement ratio. A gradient boosting-based surrogate model, trained on experimental mix data, predicts compressive strength. The modified particle swarm optimization (PSO) algorithm accommodates multiple targets in parallel, allowing solution sharing across target groups. MT-PSO is compared with a repeated PSO (R-PSO) strategy that optimizes each target separately, both minimizing the absolute error between predicted and desired strengths. Across 30 independent trials, MT-PSO consistently achieves lower mean errors, smaller deviations, and faster convergence, often reaching R-PSO's final accuracy within only a few iterations. Moreover, MT-PSO requires over 85% fewer fitness evaluations. These results demonstrate the superior accuracy, robustness, and computational efficiency of MT-PSO for multi-target optimization problems.

Keywords: multi-target optimization, mix design optimization, particle swarm optimization, surrogate modeling, gradient boosting

1. Introduction

Concrete mix proportioning is inherently an optimization process. The goal is to achieve specific performance targets while satisfying technical and practical constraints [1]. Conventional mix design approaches typically link a single target compressive strength to a single water–cement (*w*/*c*) ratio, offering limited flexibility in addressing diverse design needs [2-3]. With the emergence of artificial intelligence (AI), surrogate models have been developed to capture complex, nonlinear relationships between mix components and compressive strength. These models reveal that a given strength can be achieved through multiple feasible combinations of mixture parameters or *w*/*c* values [4-5]. To translate such predictive capabilities into practical design options, optimization algorithms are needed. These algorithms invert the surrogate model to identify the mix components for a desired strength target.

Optimization plays a central role in engineering and AI, encompassing continuous, discrete, constrained, and unconstrained problem types [6-10]. Classical methods such as steepest descent and conjugate gradient are effective for smooth and differentiable problems [11-13]. Meanwhile, linear and dynamic programming provide solutions for structured or

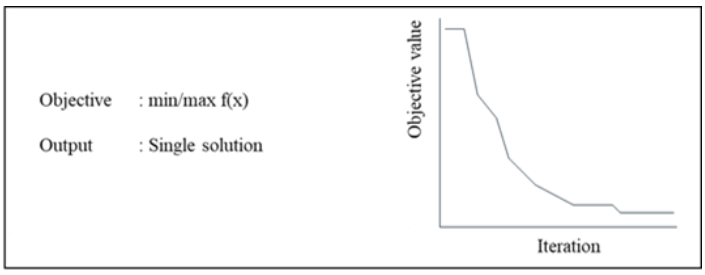
* Corresponding author. E-mail address: malikmushthofa@students.undip.ac.id

recursively decomposable tasks [14-18]. Engineering problems are increasingly nonlinear and computationally demanding. Therefore, metaheuristic algorithms gain due to their global search capabilities and flexibility [19-23]. These include simulated annealing (SA), particle swarm optimization (PSO), ant colony optimization (ACO), and grey wolf optimizer (GWO). Multi-objective optimization (MOO) methods, such as non-dominated sorting genetic algorithm II (NSGA-II) and multi-objective optimization particle swarm optimization (MO-PSO), extend these concepts to trade-off analyses among conflicting objectives [24-25]. A comparison of existing optimization paradigms with respect to their objective and target formulation is provided in Table 1.

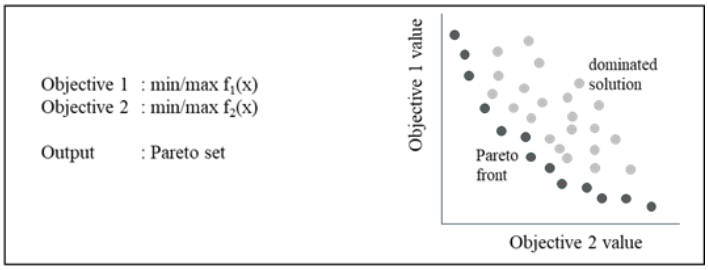
Table 1 Comparison of existing metaheuristic optimization paradigms and the proposed multi-target formulation

Optimization paradigm	Representative algorithms	Target / Objective handling
Classical (deterministic) optimization	Gradient descent,LP, NLP, DP	Optimizes a single objective with explicit constraints toward a single optimum
Metaheuristic (single-objective)	PSO,ACO, SA, GWO	Optimizes a single objective with a single predefined target
Multi-objective optimization (MOO)	NSGA-II, MOPSO, SPEA2	Simultaneously optimizes multiple conflicting objectives without explicit target values
Many-objective optimization	NSGA-III, MOEA/D	Extension of MOO for high-dimensional objective spaces
Multi-target optimization (this study)	MT-PSO (proposed)	Simultaneously optimizes multiple predefined target values for the same objective variable, which has not been systematically formulated in previous metaheuristic frameworks

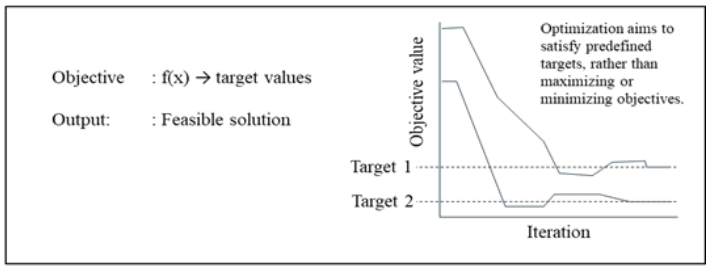
Fig. 1 illustrates the conceptual differences among three optimization paradigms. Single-objective optimization (SOO) focuses on iteratively optimizing a single objective function until convergence to a global optimum. In contrast, multi-objective optimization (MOO) simultaneously optimizes multiple conflicting objectives. This leads to a set of pareto optimal solutions that represent trade-offs rather than a single optimal solution. Multi-target optimization (MTO) differs fundamentally from both approaches by aiming to satisfy predefined target values; instead of extremizing objectives, the optimization process emphasizes convergence toward specified targets that reflect engineering design requirements.



(a) Single-Objective Optimization (SOO)



(b) Multi-Objective Optimization (MOO)



(c) Multi-Target Optimization (MTO)

Fig. 1 Conceptual comparison of optimization paradigms

An important distinction exists between objectives and targets. Objectives specify general directions for improvement, whereas targets define exact values the solution must achieve. In practical engineering applications, including concrete mix proportioning, designers often require solutions for multiple predefined target outputs rather than trade-off relationships among objectives. Although these problems are structurally multi-target in nature, existing studies overwhelmingly rely on repeatedly applying SOO algorithms to each target independently.

This repeated-SOO approach has three major limitations. First, computational cost increases linearly with the number of targets, especially when surrogate model evaluations are expensive. Second, the lack of a cross-target exploration mechanism prevents potentially valuable information sharing among optimization runs. Third, results obtained from separate SOO executions lack a unified search framework, which complicates comparison and consistency [26-27]. Existing MOO frameworks are not designed to address this gap, as they focus on simultaneous optimization of conflicting objectives rather than satisfying multiple independent targets. Consequently, there remains a methodological gap in the development of a coordinated multi-target optimization framework. This framework is required to efficiently generate mix designs corresponding to multiple predefined strength targets.

To address this gap, the present study integrates AI surrogate modeling with a novel multi-target optimization strategy. Four machine learning models, including deep neural network (DNN), random forest (RF), gradient boosting (GB), and support vector machine (SVM), are trained and hyperparameter-tuned using PSO to predict compressive strength from mix parameters. Building upon these models, a multi-target particle swarm optimization (MT-PSO) algorithm is introduced. Inspired by coordinated team-based operations, this approach deploys multiple particle ranges to indicate the squads, each assigned to a different target strength. These squads operate in parallel but remain connected through a central command mechanism that facilitates information sharing, enabling collective convergence across targets. By coupling AI-driven surrogate models with coordinated swarm intelligence, the proposed method aims to improve accuracy, accelerate convergence, and enhance computational efficiency compared with repeated SOO approaches.

The remainder of this paper is structured as follows. Section 2 presents the research significance and contextual motivation for the study. Section 3 details the methodology, including the conceptual inspiration, problem formulation, and the proposed algorithmic framework. Section 4 describes the evaluation setup, covering the surrogate model for concrete mix proportion optimization, the repetitive PSO procedure, and the MT-PSO implementation. Section 5 reports and discusses the results. The discussion covers accuracy assessment, convergence quality, convergence behavior across different target counts, computational effort analysis, and optimization outcomes. Finally, Section 6 provides the conclusions and outlines potential directions for future work.

2. Research Significance

The proposed framework advances optimization through group specialization and adaptive solution sharing. Each subpopulation is assigned to an independent target, enabling parallel searches and selective information exchange among squads. The design improves scalability, reduces unnecessary interference, and enhances convergence quality. Inspired by search-and-rescue operations, the mechanism demonstrates how decentralized coordination can accelerate the search process while maintaining task-specific focus.

To support this framework, an AI-based surrogate model is developed to provide rapid and reliable evaluations. Four models, DNN, RF, GB, and SVM, are constructed and tuned using PSO. Model performance is assessed with the coefficient of determination (R^2), mean squared error (MSE), mean absolute percentage error (MAPE), and root mean squared error (RMSE). GB consistently achieved the highest accuracy and is therefore selected as the surrogate prediction model for the optimization process.

In summary, the significance of this research lies in the integration of (i) a rigorously developed surrogate model, with GB identified as the most accurate predictor, and (ii) a novel team-based optimization framework with adaptive solution sharing. Together, these components provide a robust, efficient, and scalable strategy for addressing engineering problems with multiple predefined targets.

3. Methodology

This section details the methodology used to develop the proposed Multi-Target PSO framework. The method extends the classical PSO algorithm by incorporating a cooperative sharing mechanism to efficiently handle multiple predefined targets. The section first introduces the algorithm inspiration, followed by problem modeling, algorithm implementation, and experimental configuration relevant to engineering optimization problems.

3.1. Algorithm Inspiration

The conceptual illustration in Fig. 2 provides an intuitive analogy for the information-sharing mechanism adopted in the proposed algorithm. The inspiration is drawn from coordinated search operations, where multiple swarms operate in parallel to locate predefined targets while exchanging information through a centralized coordination framework. Each swarm explores its assigned search region independently. Meanwhile, the main algorithm collects and updates the best solutions from all swarms. It then redistributes this information to guide subsequent searches.

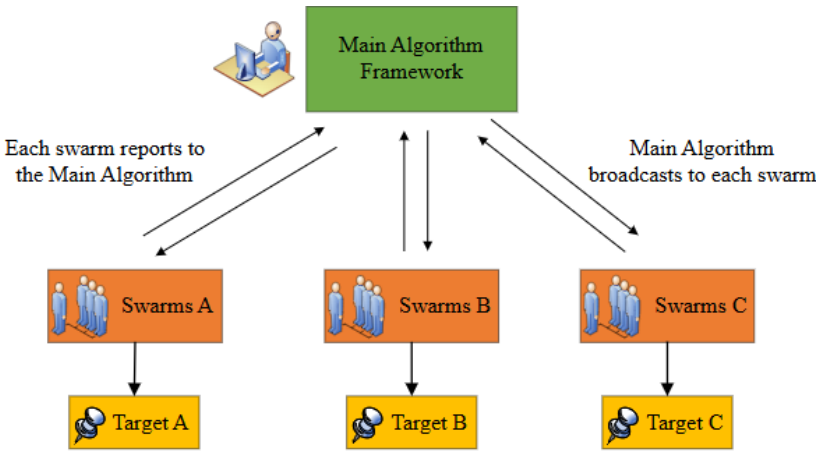


Fig. 2 The conceptual analogy of the sharing mechanism

This coordination strategy closely aligns with the fundamental principles of PSO. In PSO, each particle updates its position based on its own historical best position (personal best, *pbest*) and the best position discovered by the swarm (global best, *gbest*). In the proposed framework, individual swarms follow similar update rules, guided by both local exploration experience and globally shared information maintained by the main algorithm framework. This dual influence promotes efficient convergence while preserving sufficient exploration diversity.

The inspiration underlying this mechanism can be summarized as follows:

- Multiple swarms are assigned to different predefined targets, reflecting the structure of multi-target optimization problems.
- Global information sharing accelerates convergence by reducing redundant exploration across swarms.
- The gbest-to-pbest interaction facilitates cooperative search behavior without altering the classical PSO update dynamics.

Based on this inspiration, the classical PSO framework is extended to enable simultaneous optimization toward multiple predefined targets through cross-swarm information sharing. The resulting algorithm, termed MT-PSO, is designed to efficiently generate multiple target-specific solutions within a unified and cooperative optimization process.

3.2. Problem Modeling

To formally describe the proposed multi-target optimization strategy, a set of predefined targets is considered as $T = \{T_1, T_2, \dots, T_S\}$, where each T_S corresponds to an independent target value to be optimized by a dedicated optimization swarm $s \in \{1, \dots, S\}$. Each swarm aims to minimize the absolute error between the predicted output $\hat{y}(x)$ from a surrogate model and its corresponding target T_S, resulting in the following fitness function:

$$f_S(x) = | \hat{y}(x) - T_S | \tag{1}$$

Here, $x \in \mathbb{R}^n$ denotes a candidate solution vector constrained within the bounds $x \in [l, u]$, where l and u are the lower bound and upper bound derived from the minimum and maximum value of each feature, respectively.

Each swarm applies a standard PSO routine to minimize its own fitness function. The velocity and position updates for particle *i* in swarm *s* at iteration *t* are governed by:

$$v_i^{(t+1)} = w \cdot v_i^{(t)} + c_i \cdot r_i \cdot (p_i^{best}(T_s) - x_i^{(t)}) + c_2 \cdot r_2 \cdot (g_s^{best} - x_i^{(t)}) \tag{2}$$

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)} \tag{3}$$

where *w* is the inertia weight, c_1 and c_2 are the acceleration coefficients, and $r_1, r_2 \sim U(0,1)^n$ are vectors of random values. The terms p_i^{best} and g_s^{best} represent the personal best and global best positions, respectively. Unlike the original PSO model, in which there is only a single global-best solution, the proposed model maintains multiple swarm-specific global best solutions $g_1^{best}, \dots, g_s^{best}$, each guiding its respective target search.

At the end of the process, each swarm outputs its optimal solution:

$$x_s^{best} = \arg \min_{x_i \in \text{swarm s}} f_s(x_i) \tag{4}$$

The final result is a set of optimized solutions $\{x_1^{best}, \dots, x_s^{best}\}$, each tailored to meet a specific target. In addition, a solution-sharing mechanism is applied across swarms. Specifically, promising solutions from one swarm are evaluated against the target of another squad. A solution from swarm *j* is adopted by swarm *s* only if its fitness value improves the global best of swarm *s*:

$$f_s(x_j) < gbest_fit(s) \tag{5}$$

This ensures that only beneficial solutions are transferred, enabling cross-swarm knowledge sharing while maintaining optimization efficiency.

3.3. Algorithmic Framework and Parameter Setting

This section describes the proposed Multi-target PSO algorithm designed to address simultaneous optimization across multiple predefined targets. Each target is assigned a dedicated swarm that evolves independently using classical PSO dynamics while intermittently sharing information with other swarms to enhance convergence. The flowchart (Fig. 3) and pseudocode (Algorithm 1) summarize the algorithm structure.

Algorithm 1 Multi-target Particle Swarm Optimization (MT-PSO)

Initialization:
- *Input*:
  - Variable bounds (l, u)
  - Target set $T = \{T_1, T_2, \dots, T_S\}$ (one target per swarm)
  - Number of swarm S
  - Number of particles N
  - Maximum iterations T_{max}
  - PSO parameters: w, c_1, c_2
  - Constraints
  - Surrogate prediction model $\hat{y}(x)$
- **Initialize positions and velocities**:
  FOR each swarm s = 1 to S:
    FOR each particle i = 1 to N:
      REPEAT
        Randomly generate position x_i within bounds
        Apply constraints
        Check feasibility
      UNTIL feasible
      Evaluate fitness for swarm s:
      $f_s(x_i) = |\hat{y}(x_i) - T_s|$
      Set personal best $pbest_{i,s} = x_i, pbest_{fit(i,s)} = f_s(x_i)$
      IF $f_s(x_i) < gbest_fit(s)$ THEN
      Update $gbest_s = x_i, gbest_{fit(s)} = f_s(x_i)$

Main Loop:
FOR iteration t = 1 to T_{max}:
  FOR each swarm s = 1 to S:
    FOR each particle i = 1 to N:
      REPEAT
        Generate random vectors $r_1, r_2 \sim U(0,1)^n$
        Update velocity:
$$v_{i,s} \leftarrow w \cdot v_{i,s} + c_1 \cdot r_1 \cdot (p_{i,s}^{best} - x_{i,s}) + c_2 \cdot r_2 \cdot (g_s^{best} - x_{i,s})$$
        Update position:
        $x_{i,s} \leftarrow x_{i,s} + v_{i,s}$
        Apply bounds and constraints
      UNTIL feasible

      Evaluate fitness for all swarm:
      $f_{ss}(x_{i,s}) = |\hat{y}(x_{i,s}) - T_{ss}|, \qquad \forall ss = 1, \dots, S$

      Sharing mechanism:
        FOR each swarm index ss = 1 … S:
          IF $f_{ss}(x_{i,s}) < gbest_fit(ss)$, THEN
            Update $gbest_{ss} = x_{i,s}, \quad gbest_fit(ss) = f_{ss}(x_{i,s})$
          IF $f_{ss}(x_{i,s}) < pbest_{fit(i,ss)}$, THEN
            Update $pbest_{i,ss} = x_{i,s}, \quad pbest_fit(i,ss) = f_{ss}(x_{i,s})$

Output:
FOR each swarm s, return optimal solution and fitness:
$x_s^{best} = gbest_s, \quad f_s(x_s^{best}) = gbest_fit(s)$
Final result is the set of swarm -specific solutions:
$\{x_1^{best}, x_2^{best}, \dots, x_S^{best}\}$

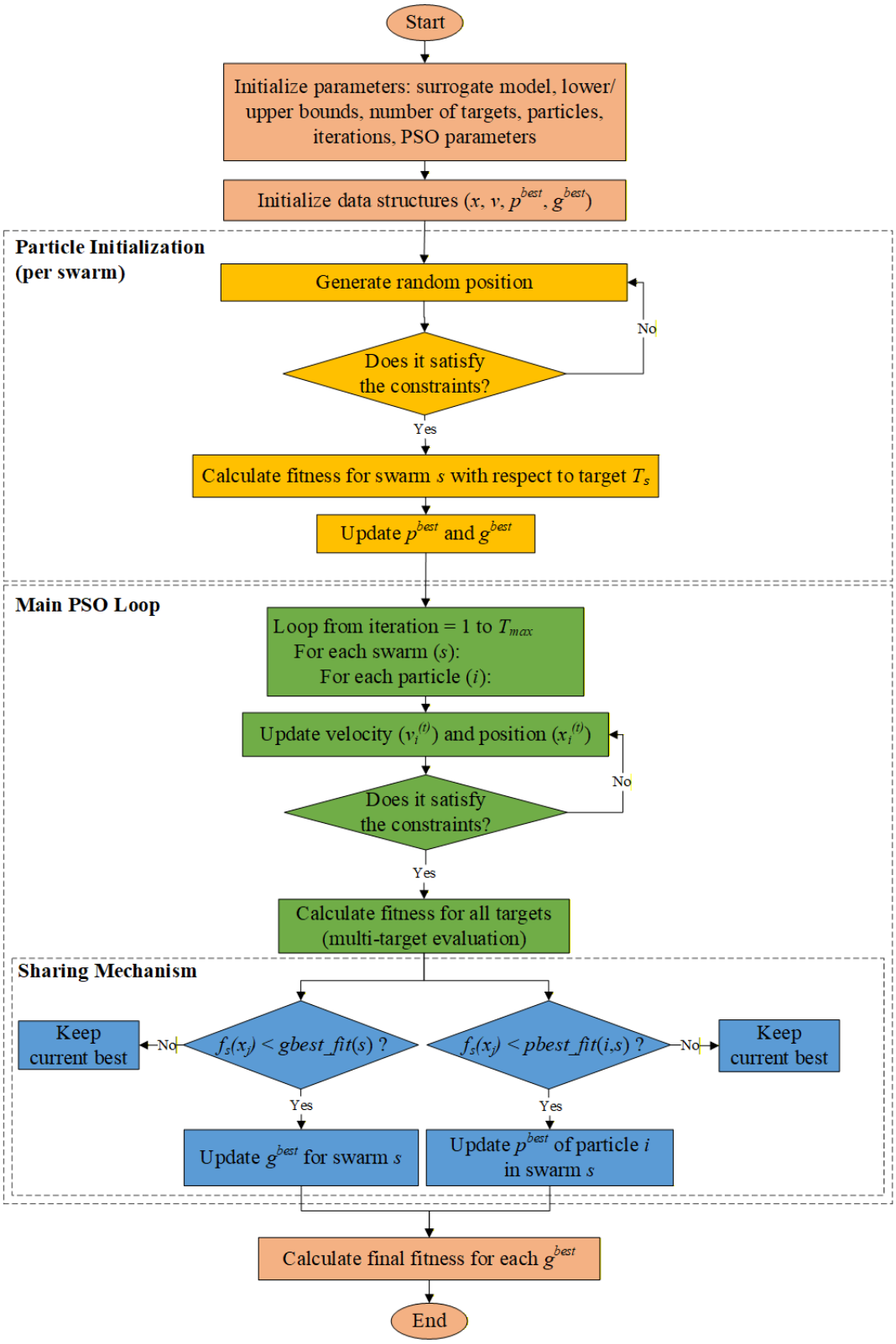


Fig. 3 The conceptual analogy of the sharing mechanism

It should be emphasized that MT-PSO is not intended for training or benchmarking predictive models. Instead, the algorithm operates on pre-trained surrogate models that represent the relationship between mix design variables and concrete performance. In this framework, the surrogate model serves as an evaluator, enabling rapid assessment of candidate solutions during optimization. The primary role of MT-PSO is to solve a multi-target design optimization problem by coordinating multiple swarms toward predefined target values. This separation between modeling and optimization ensures that the proposed method remains model-agnostic and focused on optimization efficiency and target satisfaction.

Table 2 summarizes the parameter settings adopted in the proposed MT-PSO algorithm, together with their functional roles and corresponding performance implications. These parameters govern key aspects of the search process, including swarm interaction, exploration-exploitation balance, and information sharing across targets. Rather than introducing additional experimental scenarios, the selected values reflect commonly accepted PSO configurations and are chosen to ensure stable convergence and computational efficiency. Presenting this information in a compact tabular form facilitates reproducibility. It provides insight into how each parameter influences algorithmic behavior and optimization performance.

Table 2 Algorithm parameter settings and their performance implications

Category	Parameter	Symbol	Functional Role	Performance Implication
Swarm structure	Number of swarm groups	S	Assigns one swarm per target	Enables simultaneous multi-target optimization
Swarm structure	Particles per swarm	N	Controls search diversity	Balanced exploration with limited computational cost
Velocity update	Inertia weight	w	Balances exploration vs exploitation	Accelerates early exploration and avoids premature convergence
Velocity update	Cognitive coefficient	c_1	Guides particles toward personal best	Enhances local refinement capability
Velocity update	Social coefficient	c_2	Guides particles toward global best	Promotes swarm-level convergence
Constraint handling	Fixed constraint	l, u	Enforces material proportion	Reduces infeasible solutions, shrinks search space
Constraint handling	Fixed variable	–	Fixes design parameter	
Termination	Maximum iterations	T_{max}	Stops optimization	Ensures convergence stability
Information sharing	Cross-swarm sharing	–	Exchanges best solutions across swarms	Reduces redundant fitness evaluations and improves accuracy
Optimization scope	Target handling	T	Defines fitness function	Focuses convergence toward predefined target values

The selected configuration reflects a trade-off between computational efficiency and optimization accuracy, as further validated by the experimental results. Specifically, the adopted parameter values were found to provide sufficient exploration capability in the early search stages while enabling rapid convergence as the algorithm progresses. This balance is particularly important in multi-target optimization. Excessive exploration increases computational cost, whereas overly aggressive exploitation may lead to premature convergence for certain targets.

4. Evaluation Setup

The concrete mix design methods predominantly rely on the water-to-cement (*w/c*) ratio as the main determinant of compressive strength [3]. A single target strength is typically achieved using a fixed w/c ratio. In this study, a surrogate model is developed based on concrete proportion data and compressive strength. The model is then reversed using optimization, allowing material proportions to be generated for each target strength. The purpose of this model development is to determine concrete proportions for a given target strength without being restricted to a specific w/c ratio. For this study, a *w/c* ratio of 0.5 is selected for optimization (considering workable concrete conditions), with target strengths of 20, 25, 30, 35, and 40 MPa, representing the range of normal-strength concrete. This model development also serves as an evaluation setup to test whether Multi-Target PSO can outperform Repetitively Running PSO.

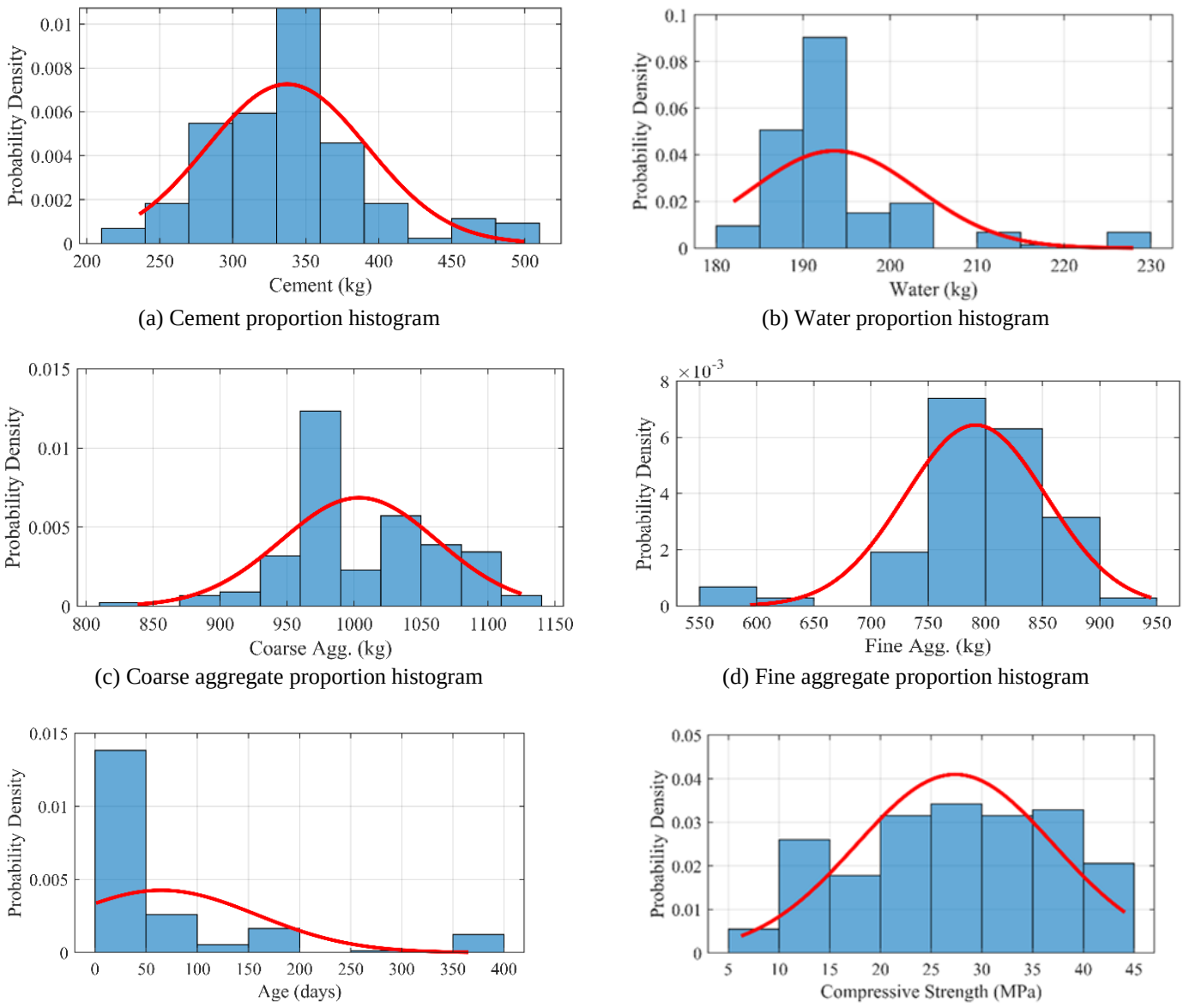
4.1. Surrogate Model for Concrete Mix Proportion Optimization

The traditional concrete data is collected from a study conducted by Yeh [28]. The final dataset includes cement, water, proportions of coarse and fine aggregate, and age as an additional parameter. A statistical analysis is then performed on the dataset, including calculations of the mean, median, standard deviation, minimum and maximum values, and data range, as presented in Table 3.

Table 3 Properties data parameter input and output

Parameters	Mean	Median	StdDev	Min	Max	Range
Portland Cement (kg)	323.78	310.00	54.55	236.00	500.00	264.00
Water (kg)	193.03	192.00	8.49	182.00	228.00	46.00
Coarse Aggregate (kg)	995.84	969.00	53.40	838.40	1125.00	286.60
Fine Aggregate (kg)	805.84	812.00	60.85	594.00	945.00	351.00
Age (day)	75.41	28.00	103.71	1.00	365.00	364.00
Compressive Strength (MPa)	27.05	27.19	9.35	6.27	44.09	37.82

The distribution of each parameter is presented using histograms, as shown in Fig. 4. A correlation matrix illustrating the relationships between variables is provided in Fig. 5. The results indicate no autocorrelation among the input parameters, as none of the correlation values approach 1.00. Additionally, the most influential factor on compressive strength is the concrete's age. Among the material proportions, cement has the greatest impact on compressive strength. The dataset is split into 80% for training and 20% for testing. Before training and testing, the data are standardized. A separate validation dataset is not created; instead, holdout validation is applied using an early stopping technique.



(a) Cement proportion histogram

(b) Water proportion histogram

(c) Coarse aggregate proportion histogram

(d) Fine aggregate proportion histogram

(e) Concrete age histogram

(f) Compressive strength histogram

Fig. 4 Histogram for input and output parameters with normal distribution

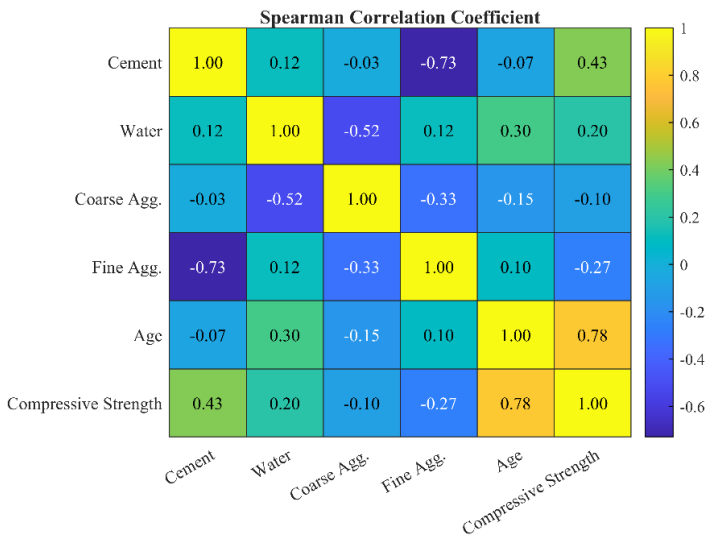


Fig. 5 Spearman correlation matrix of each parameter

This study employs 4 types of hybrid machine learning, namely DNN, RF, SVM, and GB, with the PSO employed for hyperparameter tunering. Table 4 shows the hyperparameters of the implemented AI model. The ranges indicate the hyperparameter values optimized using the PSO algorithm [29]. The PSO tuned 2 dimensions of hyperparameters for each AI model. To measure the predictive power of the models, four measures of error were used: coefficient of determination (R^2), mean squared error (*MSE*), root mean squared error (*RMSE*), and mean absolute percentage error (*MAPE*). Each provides different information regarding model performance.

Table 4 The model hyperparameters

Model	Hyperparameter	Parameter value	Optimal value
DNN-PSO	Layer 1	10 – 20	20
	Layer 2	20 – 30	25
	Layer 3	10 – 25	16
	Activations	relu	relu
	IterationLimit	1000	1000
	StepTolerance	1e-10	1e-10
	LossTolerance	1e-10	1e-10
RF-PSO	NumLearningCycles	100 – 3000	2214
	LearnRate	1e-4 – 1e-1	0.0439
	Method	Bag	Bag
GB-PSO	NumLearningCycles	100 – 3000	198
	LearnRate	1e-4 – 1e-1	6.5781e-02
	Method	LSBoost	LSBoost
SVM-PSO	BoxConstraint	1e-3 – 1e3	531.4651
	KernelScale	1e-2 – 1e3	1.6389
	KernelFunction	rbf	rbf

In this study, R^2 is selected as the primary fitness objective for hyperparameter optimization instead of directly minimizing the mean squared error (Fig. 6). Unlike MSE, which evaluates absolute prediction errors and is sensitive to data scale and outliers, R^2 provides a normalized measure of the explained variance and directly reflects the surrogate model's ability to capture the relationship between mix design variables and compressive strength. Notably, error-based metrics such as MSE remain important for evaluating absolute prediction accuracy; therefore, MSE and RMSE are reported in the results section to provide a complementary assessment of model performance. R^2 is better suited as a fitness objective in PSO-based surrogate modeling because it emphasizes explanatory power, stability, and generalization, which are more critical for optimization-driven engineering design than minimizing pointwise error alone.

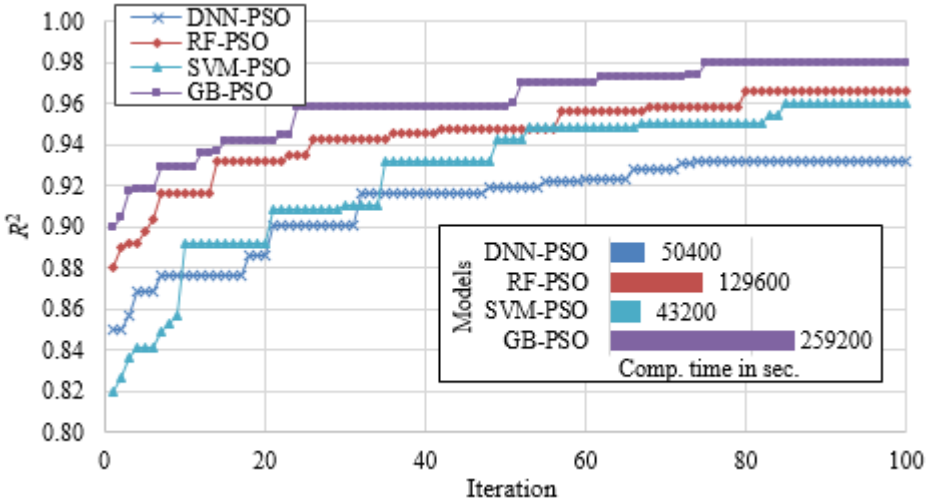


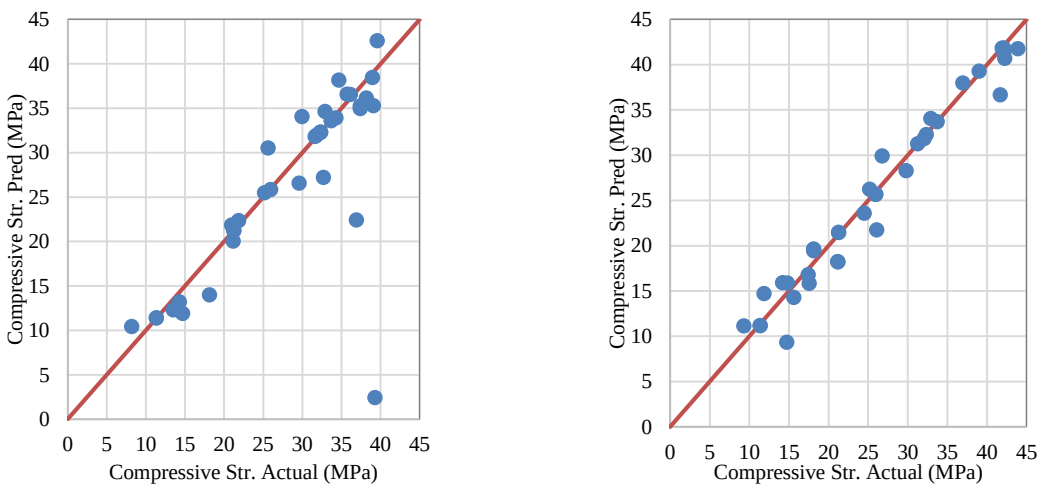
Fig. 6 Convergence curve and computational time comparison among hybrid models

While the DNN-PSO is the best on training for all metrics (a reflection of good learning capability), it suffers from serious overfitting, as reflected in its much worse R^2, *MSE*, *RMSE*, and *MAPE* scores on the testing phase. As shown in Table 5, the GB-PSO model, on the other hand, provides a more stable training and testing performance. In testing, GB-PSO achieved better results than all the models in three out of the four metrics: R^2 (0.983), *MSE* (2.231), *RMSE* (1.493 MPa), and *MAPE* (0.035), indicating its superior generalization ability to unseen data. Although both RF-PSO and SVM-PSO were stable with high performance, neither of them performs better than GB-PSO in any metric.

Table 5 Error performance of each model

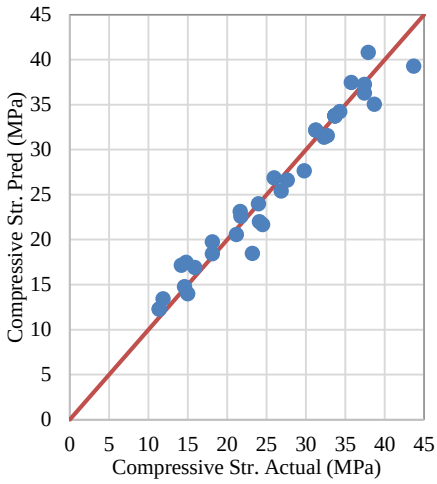
Model	Training				Testing			
	R^2	*MSE*	*RMSE*	*MAPE*	R^2	*MSE*	*RMSE*	*MAPE*
DNN-PSO	0.997	0.214	0.462	0.009	0.932	6.503	2.550	0.060
RF-PSO	0.989	0.942	0.971	0.027	0.966	3.818	1.954	0.068
SVM-PSO	0.977	2.130	1.459	0.055	0.960	3.263	1.807	0.059
GB-PSO	0.997	0.247	0.497	0.009	0.983	2.231	1.493	0.035

Fig. 7 shows the scatter plot of testing data for each developed model. DNN-PSO, although powerful, struggles with small datasets due to its high complexity and large number of trainable parameters. This often leads to overfitting, where the model memorizes the data instead of learning general patterns. In contrast, models like GB-PSO, RF-PSO, and SVM-PSO are better suited for small datasets. GB-PSO and RF-PSO use ensemble techniques to reduce variance, while SVM-PSO constructs optimal decision boundaries using limited data.

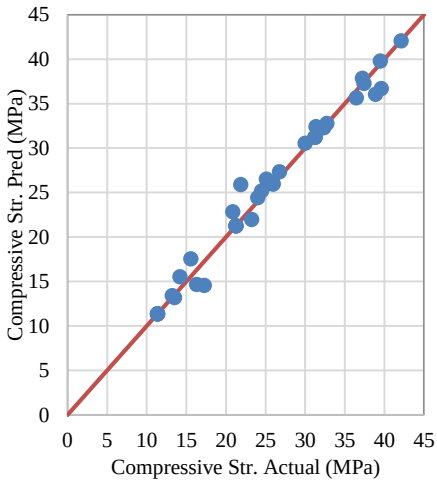


(a) DNN-PSO (b) RF-PSO

Fig. 7 Scatter plot testing datapoints



(c) SVM-PSO



(d) GB-PSO

Fig. 7 Scatter plot testing datapoints (continued)

Among these, GB-PSO is selected as the best model due to its strong generalization and balanced performance on the test set. However, GB-PSO generally requires the longest computation time because it builds trees sequentially, with each tree correcting the errors of its predecessor. This iterative, gradient-based process prevents parallelism and increases training time.

4.2. Repetitive PSO Running and Multi-Target PSO Result

After identifying the best surrogate model, which is the Gradient Boosting (GB) model, the optimization process continues using optimization algorithms. This process aims to determine the optimal material proportions based on the target compressive strength using the surrogate model. At this stage, the optimization is performed using two methods: the first is the standard PSO, which executes repeatedly for each target (R-PSO); and the second is the MT-PSO, which is executed once in parallel for all targets by activating a sharing mechanism.

In this simulation, the water-to-cement ratio (*w/c* ratio) is constrained to a fixed value of 0.5. The optimization simulation is conducted for target strengths ranging from 20 to 40 MPa with 5 MPa intervals. The objective function used in this simulation is to minimize the error or fitness value using absolute error. Each target optimization is run 30 times using the hyperparameters as presented in Table 6, and the statistical results (including mean, standard deviation, best, and worst values) are presented in Table 7.

Table 6 Optimization hyperparameter

Optimization algorithm	Hyperparameter variable	Hyperparameter value
R- PSO	particles number	10
	iterations number	50
	w; c_1; c_2	0.9; 1.5; 1.5
MT-PSO	squads number	5; [20 25 30 35 40]
	particles number	10
	iterations number	3; 5; 10; 20; 50
	w; c_1; c_2	0.9; 1.5; 1.5

Table 7 Fitness error statistic

Target (MPa)	Repeatedly Running PSO		Multi-Target PSO									
	T_{max}= 50		T_{max}= 3		T_{max}= 5		T_{max}= 10		T_{max}= 20		T_{max}= 50	
	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max
20	1.128 (1.153)	0.532-4.037	0.825 (0.697)	0.532-3.893	0.719 (0.611)	0.532-3.893	0.549 (0.064)	0.532-0.784	0.557 (0.077)	0.532-0.784	0.532 (0.000)	0.532-0.532

Target (MPa)	Repeatedly Running PSO		Multi-Target PSO									
	T_{max}= 50		T_{max}= 3		T_{max}= 5		T_{max}= 10		T_{max}= 20		T_{max}= 50	
	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max	Mean (Std.)	Min -Max
25	0.285 (0.334)	0.001-1.364	0.190 (0.228)	0.001-0.805	0.109 (0.108)	0.001-0.446	0.030 (0.043)	0.001-0.145	0.010 (0.015)	0.001-0.055	0.002 (0.003)	0.001-0.012
30	0.082 (0.113)	0.004-0.406	0.046 (0.047)	0.004-0.185	0.032 (0.031)	0.004-0.102	0.016 (0.015)	0.001-0.057	0.011 (0.016)	0.001-0.090	0.004 (0.003)	0.001-0.013
35	0.197 (0.257)	0.002-0.959	0.115 (0.135)	0.010-0.578	0.119 (0.174)	0.017-0.734	0.048 (0.040)	0.009-0.190	0.038 (0.027)	0.001-0.112	0.016 (0.010)	0.000-0.029
40	0.039 (0.031)	0.003-0.085	0.042 (0.031)	0.003-0.085	0.032 (0.023)	0.003-0.080	0.017 (0.015)	0.003-0.068	0.011 (0.007)	0.003-0.019	0.009 (0.007)	0.003-0.018

5. Results and Discussion

This section presents and discusses the performance of the proposed MT-PSO framework from both predictive and optimization perspectives. The results are organized to assess model accuracy, convergence behavior, computational efficiency, and optimization outcomes under multiple predefined target strengths. Quantitative analyses are supported by statistical evaluations and convergence assessments to ensure robustness and reproducibility. The discussion further interprets the results in the context of engineering applicability, highlighting the advantages and limitations of the proposed approach relative to conventional optimization strategies.

5.1. Accuracy

The mean fitness values clearly demonstrate that MT-PSO consistently outperforms R-PSO across all target pressures and iteration counts. Even after only 3 iterations, MT-PSO achieves mean fitness values significantly lower than R-PSO's results after 50 iterations (Table 5). For instance, at the 20 MPa target, R-PSO's mean error is 1.128 MPa, while MT-PSO achieves 0.825 MPa at 3 iterations (a reduction of approximately 26.8%), and this drops further to 0.532 MPa at 50 iterations (a reduction of 52.8% compared to R-PSO). Similar accuracy gains are observed across all targets, with reductions in mean error ranging from roughly 40%–95% depending on the iteration count and target. This indicates that MT-PSO's multi-target knowledge sharing allows for rapid convergence toward more precise solutions, even with minimal iterations.

The best fitness values for MT-PSO are remarkably stable and close to zero from the earliest iterations, frequently reaching 0.001 MPa, and remain consistent up to 50 iterations. This suggests that MT-PSO quickly finds near-optimal solutions and maintains them across runs. In contrast, while R-PSO occasionally achieves low best values, these are not consistently maintained across all targets, pointing to a higher dependency on stochastic variation rather than systematic accuracy.

The standard deviation (std) values further confirm MT-PSO's superior accuracy and stability. At all targets, MT-PSO's std values are much lower than those of R-PSO, often by more than 70%, especially at higher iterations. For example, at 25 MPa, R-PSO's std is 0.334, while MT-PSO achieves just 0.015 at 20 iterations and 0.003 at 50 iterations. This low variability indicates that MT-PSO's solutions are not only more accurate on average but also more consistent, reducing the reliance on chance in obtaining high-quality solutions.

Overall, these results confirm that MT-PSO delivers superior accuracy over R-PSO across all targets, and crucially, it achieves these gains in far fewer iterations. The method's ability to combine rapid convergence with consistent precision highlights its suitability for optimization problems where both accuracy and efficiency are critical.

5.2. Convergence Quality

The convergence patterns in the data reveal that MT-PSO reaches high-quality solutions significantly faster than R-PSO. Even after only 3 iterations, MT-PSO achieves mean fitness values lower than R-PSO's performance after a full 50 iterations in most target cases. For example, at the 30 MPa target, MT-PSO achieves a mean fitness of 0.046 in 3 iterations, while R-PSO, even after 50 iterations, remains at 0.082 (Table 7). This indicates that MT-PSO's search process is inherently more efficient, exploiting inter-target information sharing to accelerate convergence toward the optimal region of the solution space.

Another strong indicator of rapid convergence is the steep drop in mean fitness within the first few MT-PSO iterations. Between iterations 3 and 10, MT-PSO's mean fitness often improves by more than 40%–60% (Table 8), after which the improvement rate slows but continues steadily toward near-optimal values. By iteration 20, MT-PSO's performance is already very close to the iteration 50 results, demonstrating that most of the convergence is achieved early in the process. In contrast, R-PSO shows much slower improvement rates, requiring the full 50 iterations to reach results that are still inferior to MT-PSO's early-stage outcomes.

Table 8 MT-PSO fitness improvement compared to R-PSO

Target (MPa)	MT-PSO Fitness Improvement				
	3 iter.	5 iter.	10 iter.	20 iter.	50 iter.
20	26.8%	36.2%	51.3%	50.6%	52.8%
25	33.3%	61.8%	89.6%	96.6%	99.3%
30	43.6%	61.0%	80.0%	86.6%	95.2%
35	41.8%	39.6%	75.6%	80.9%	92.1%
40	-8.5%	16.7%	56.2%	72.9%	76.8%

The standard deviation trend further underscores MT-PSO's strong convergence behavior. As iterations progress, MT-PSO's std values decrease rapidly, indicating that particles quickly cluster around the optimal solution and maintain this focus. This is a hallmark of healthy convergence, where the algorithm narrows its search efficiently without premature stagnation. R-PSO, however, maintains relatively larger std values even after 50 iterations. This suggests a less focused swarm and a slower consolidation of solutions.

Finally, the best fitness stability in MT-PSO (where near-optimal values are reached and maintained from the earliest iterations) confirms that convergence is both rapid and robust. This consistency means MT-PSO not only finds optimal solutions faster but also retains them without significant degradation in subsequent iterations, a performance that R-PSO struggles to match even with its longer run time.

5.3. Convergence Behavior Analysis for Different Target Numbers

The three plots represent the convergence trends of MT-PSO across different optimization scenarios with varying numbers of targets (3, 5, 7, 9, and 11) for target strengths of 20 MPa, 30 MPa, and 40 MPa. A consistent and notable pattern emerges: the more targets optimized simultaneously, the faster the algorithm converges to its minimum fitness value.

In the 20 MPa case (Fig. 8(a)), configurations with a higher number of targets (7, 9, and 11) reach the plateau region within fewer than 5 iterations, whereas configurations with fewer targets (3 and 5) require up to around 10 iterations. This acceleration in convergence is directly linked to the wider exploration coverage afforded by optimizing more targets at once. With more objectives in play, particles distribute themselves more diversely in the solution space, enabling the swarm to identify promising regions more quickly.

For 30 MPa (Fig. 8(b)), this trend is even more evident. When optimizing 9 or 11 targets, the fitness drops sharply within the first few iterations, often reaching near-optimal solutions well before iteration 10. In contrast, the 3-target configuration

maintains a higher fitness level until roughly iteration 8–10 before stabilizing. The rapid convergence for higher-target cases suggests that the multi-target sharing mechanism is especially effective at propagating useful information between targets early in the optimization process, reducing redundant exploration.

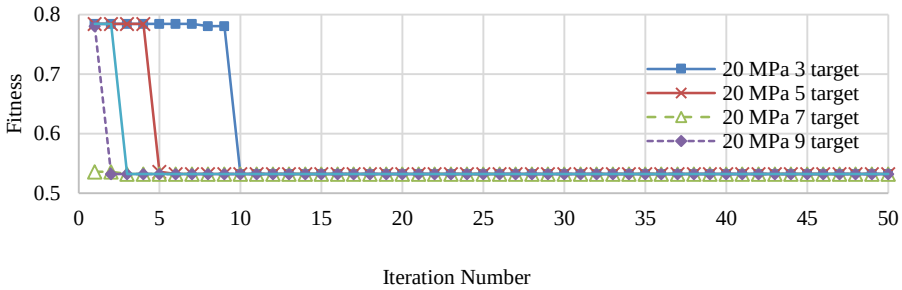
The 40 MPa scenario (Fig. 8(c)) reinforces this observation. The 11-target setup achieves its best solution in only a few iterations, with very little improvement needed thereafter. Even though all configurations eventually converge to similarly low fitness values, the path taken is markedly shorter for larger target counts. This implies that the algorithm's convergence efficiency benefits from the synergy among multiple optimization tasks, where high-quality solutions found for one target can directly guide the search for others. Overall, these results clearly illustrate that increasing the number of targets in MT-PSO enhances convergence speed due to:

(1) Broader exploration: More targets diversify the swarm's search space coverage.

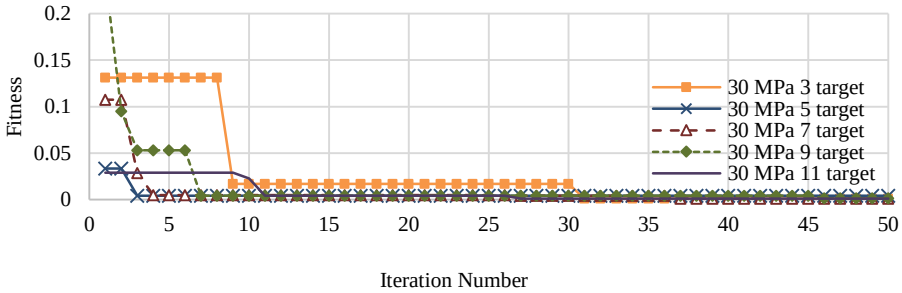
(2) Accelerated exploitation via sharing: Information sharing between targets enables quicker identification of optimal regions.

(3) Reduced stagnation risk: Multi-target interaction prevents the swarm from prematurely focusing on suboptimal zones.

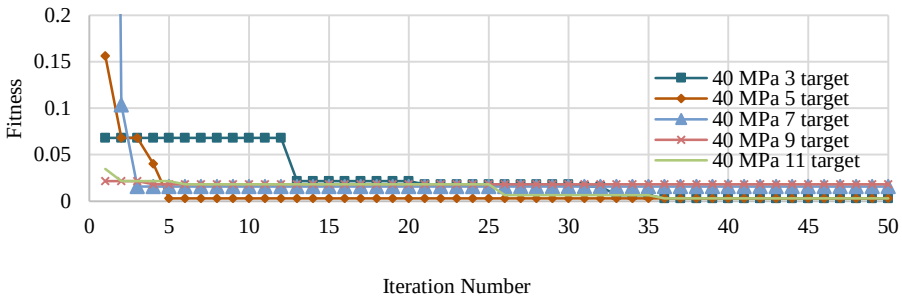
This set of graphs, therefore, not only confirms the advantage of MT-PSO over traditional single-target approaches but also shows that scaling up the number of simultaneous targets can further improve convergence behavior without compromising final solution quality.



(a) 20 MPa



(b) 30 MPa



(c) 40 MPa

Fig. 8 Convergence curve for different target numbers

Fig. 9 illustrates the convergence behavior of the optimization process across iterations, augmented with error bands that represent the variability of fitness values within the population. The solid black curves denote the mean fitness evolution, while the shaded regions indicate dispersion, reflecting exploration diversity and population stability during the process.

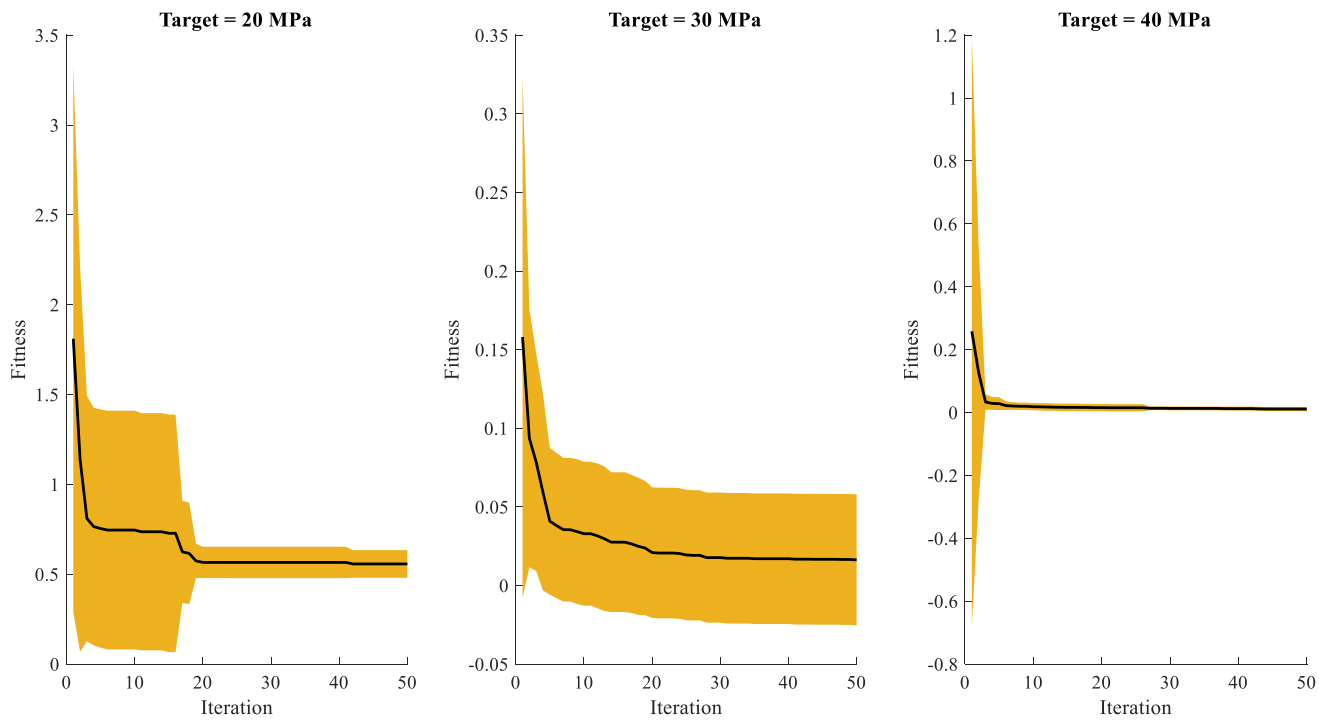


Fig. 9 Error bands curves of the MT-PSO among the target strengths

The convergence is not abrupt, indicating that diversity is reduced in a controlled manner rather than collapsing prematurely, which is a desirable property in swarm-based optimization. In the final iteration phase, the convergence curves approach a plateau, and the error bands become narrow and nearly constant. The minimal variance in this phase confirms that the swarm has reached a stable solution region, suggesting the robustness of the optimization process. The absence of oscillatory behavior or variance re-expansion indicates that the algorithm is not trapped in unstable local optima.

A comparative observation across the three subplots reveals differences in convergence dynamics and stability. The leftmost case exhibits a relatively larger residual error band at convergence, implying slightly higher solution variability and a broader basin of acceptable solutions. In contrast, the middle and rightmost cases show tighter final error bands, indicating stronger convergence consistency and higher confidence in the obtained optima. Notably, the rightmost case demonstrates the fastest variance collapse, suggesting either a smoother objective landscape or a particularly well-suited algorithm configuration for this target. Overall, the narrowing of error bands alongside monotonic fitness improvement confirms that the proposed optimization framework achieves stable and reliable convergence without sacrificing early-stage exploration capability.

5.4. Computational Effort

Based on the Fitness Function Call (FFC) data, MT-PSO demonstrates a significant computational efficiency advantage over R-PSO across all target counts (3, 5, and 7 targets) and for all tested iteration numbers (Table 9). In the scenario with 3 targets, MT-PSO requires 86%–90% fewer function calls compared to R-PSO. For example, at the 3rd iteration, MT-PSO needs only 123 FFC, whereas R-PSO requires 900 FFC, an approximate reduction of 86.33%. Even at the 200th iteration, MT-PSO maintains an efficiency gain of roughly 89.95%. This indicates that, despite longer optimization runs, MT-PSO avoids excessive function evaluations.

Table 9 Fitness function call evaluation

Target Number	Iteration Number	Fitness function call		
		MT-PSO	R-PSO	Efficiency
3	3	123	900	86.33%
	10	333	3000	88.90%
	50	1533	15000	89.78%
	200	6033	60000	89.95%
5	3	205	1500	86.33%
	10	555	5000	88.90%
	50	2555	25000	89.78%
	200	10055	100000	89.95%
7	3	287	2100	86.33%
	10	777	7000	88.90%
	50	3577	35000	89.78%
	200	14077	140000	89.95%

A similar pattern is observed for 5 targets. At low iterations (e.g., the 3rd iteration), MT-PSO needs only 205 FFC compared to 1,500 FFC for R-PSO, corresponding to approximately 86.33% savings. At the 200th iteration, the savings remain around 89.95%. This demonstrates that MT-PSO scales effectively with an increasing number of targets, maintaining high efficiency despite the added problem complexity.

For 7 targets, although the absolute number of function evaluations increases, the percentage savings remain consistent. At the 3rd iteration, MT-PSO requires 287 FFC, whereas R-PSO requires 2,100 FFC (86.33% efficiency). By the 200th iteration, MT-PSO still achieves nearly 90% savings compared to R-PSO. This consistency indicates that the multi-target sharing mechanism in MT-PSO effectively leverages information across targets, accelerating the search process without excessive function evaluations.

Overall, these results confirm that MT-PSO excels not only in accuracy but also in computational efficiency, achieving over 85% FFC savings across all target counts and iteration numbers. This makes MT-PSO a highly suitable choice for large-scale multi-target optimization problems, particularly when function evaluations are computationally expensive.

Although MT-PSO exhibits higher computational time per iteration compared to R-PSO (Table 10), this does not conflict with the observed reduction in fitness function calls. The increased runtime per iteration originates from additional coordination, target evaluation, and information-sharing mechanisms embedded in MT-PSO, which are absent in standard PSO. Importantly, fitness function calls represent only a subset of the total computational cost. Despite higher per-iteration overhead, MT-PSO achieves substantially higher optimization accuracy using significantly fewer iterations, demonstrating superior search efficiency and reduced redundancy compared to repetitive PSO executions. MT-PSO sacrifices per-iteration computational simplicity in exchange for superior information efficiency, resulting in faster effective convergence and significantly improved solution accuracy (as illustrated in Fig. 10).

Table 10 MT-PSO execution time compared to R-PSO

Variable	R-PSO	MT-PSO				
	50 iter.	3 iter.	5 iter.	10 iter.	20 iter.	50 iter.
Execution time (sec.)	109.45	39.51	57.28	104.35	275.20	640.52
Time per iteration (sec.)	2.19	13.17	11.46	10.43	13.76	12.81

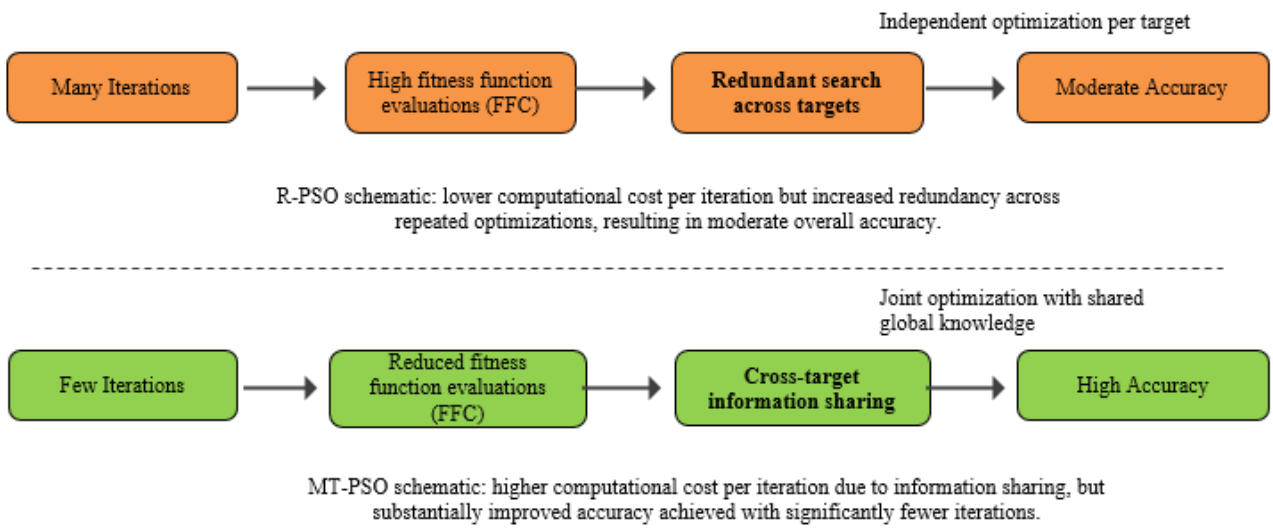


Fig. 10 Conceptual comparison between R-PSO and the proposed MT-PSO

5.5. Statistical Analysis of Optimization Performance

To evaluate the statistical stability of the optimization results, multiple independent runs are conducted for both MT-PSO and the repeated SOO-based PSO under identical parameter settings. The mean and standard deviation of the final objective values are computed for each target.

A paired-sample t-test is performed to determine whether statistically significant differences exist between the final optimization outcomes of MT-PSO and R-PSO. At a significance level of $\alpha = 0.05$, no statistically significant differences are observed for any of the evaluated target strengths. The corresponding confidence intervals include zero, indicating comparable solution quality between the two approaches (Table 11).

Table 11 The fitness data paired t-test results

Target strength data	Paired Differences					t	df	Sig. (2-tailed)
	Mean	Std. Dev.	Std. Error Mean	95% C.I. of the Difference				
				Lower	Upper			
MT-PSO_20 – R-PSO_20	-0.421	2.525	0.461	-1.3637	0.5221	-0.913	29	0.369
MT-PSO_30 – R-PSO_30	-0.016	0.159	0.029	-0.0758	0.0433	-0.559	29	0.581
MT-PSO_40 – R-PSO_40	0.006	0.071	0.013	-0.0203	0.0327	0.480	29	0.635

These results demonstrate that MT-PSO is able to achieve optimization performance equivalent to repeated SOO-based PSO in terms of final objective values, while offering additional advantages in computational efficiency and unified multi-target search capability, as discussed in the subsequent sections.

Fig. 11 presents box plots of the final optimization outcomes from 30 independent runs for MT-PSO and R-PSO across different target strength levels. For targets of 30 and 40 MPa, both methods exhibit closely aligned medians and narrow interquartile ranges, indicating comparable convergence quality and stable optimization behavior. At the lower target level of 20 MPa, although the median performances remain similar, R-PSO demonstrates a substantially wider interquartile range and several extreme outliers, reflecting increased variability across runs. In contrast, MT-PSO maintains a more compact distribution, suggesting enhanced robustness under more challenging target conditions. These distributional observations are consistent with the paired-sample t-test results, which indicate no statistically significant differences in mean performance between the two methods.

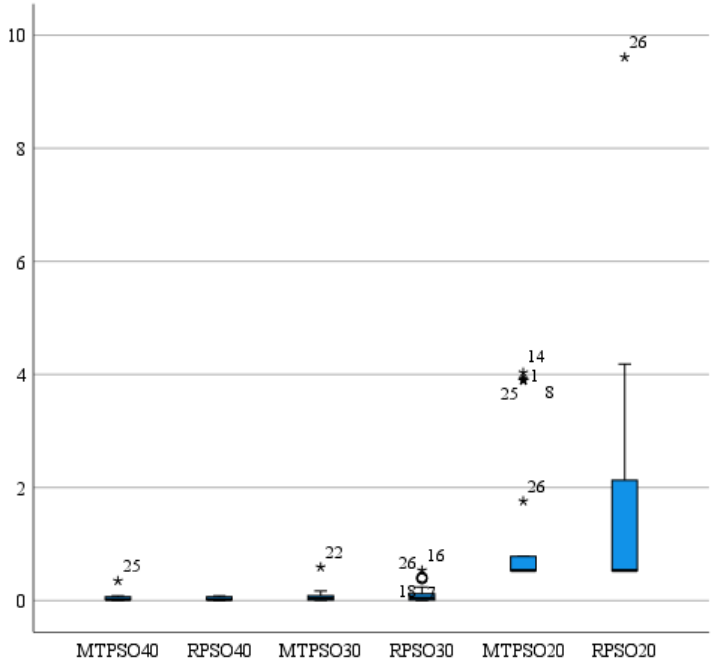


Fig. 11 Fitness comparison with boxplots chart

5.6. Optimization result

Based on the model's prediction results for five normal-strength concrete targets (as shown in Table 12), the results show that all predicted values closely match the targets, with very small fitness values (absolute error). An interesting aspect of this data is that all concrete mixtures use a constant water-to-cement ratio (*w/c* ratio) of 0.5, which is calculated from the weight proportion of cement (PC) and water (W) in each sample (e.g., 128/255 ≈ 0.5, 159/317 ≈ 0.5, etc.). This indicates that even though the w/c ratio remains unchanged, the compressive strength of the concrete can still be controlled and achieved by adjusting the composition of the other materials.

Table 12 Traditional concrete proportion optimization result, with constraint *w/c* ratio as 0.5

Target (MPa)	PC (kg)	W (kg)	CA (kg)	FA (kg)	Age (day)	Predicted Str. (MPa)	Fitness
20	255	128	1125	881	28	20.5319	0.5320
25	317	159	1105	810	28	25.0007	0.0007
30	452	226	944	770	28	29.9989	0.0011
35	384	192	939	869	28	34.9977	0.0023
40	460	230	1034	668	28	40.0030	0.0030
Note: PC = Portland cement; W = Water; CA = Coarse aggregate; FA = Fine aggregate.							

This finding contrasts with the traditional approach in mix design, in which a given target strength is typically associated with a specific *w/c* ratio, which is then set using empirical relationship curves. In this approach, it is found that by maintaining a constant *w/c* ratio (e.g., 0.5), a wide range of compressive strengths can still be flexibly obtained, as long as the other parameters are proportionally adjusted.

6. Conclusions and Future Work

This study successfully established and validated the multi-target particle swarm optimization (MT-PSO) framework. The main accomplishments and findings are summarized as follows:

(1) Challenge addressed: Repeated single-objective optimization becomes computationally inefficient when multiple predefined targets are required in a concrete mix design.

(2) Framework capability: The MT-PSO framework enables simultaneous optimization of multiple target strengths within a

unified search process.

(3) Surrogate modeling: Four machine learning models were constructed and optimized using hyperparameter tuning, with the gradient boosting model demonstrating the most stable predictive performance.

(4) Performance comparison: MT-PSO consistently achieved lower mean fitness values and faster convergence than the repeated SOO-based PSO across all target strengths.

(5) Computational efficiency: The proposed framework reduced fitness evaluations by more than 85% compared to the conventional approach.

(6) Practical implications: MT-PSO offers an efficient and robust optimization tool for multi-target concrete mix design and similar engineering applications.

(7) Flexibility: The MT-PSO framework enables flexible multi-target mix design under practical constraints and remains extensible to incorporate additional engineering considerations such as cost, feasibility, and durability.

For future work, the proposed sharing mechanism will be adapted to other swarm-based optimization algorithms to further enhance solution diversity and convergence behavior. Physics-informed surrogate models will be incorporated to improve generalization and reliability, while explainable AI techniques will be employed to interpret material–performance relationships. The framework will be extended to other concrete systems, including high-strength, recycled aggregate, and geopolymer concretes.

Acknowledgments

The authors gratefully acknowledge the financial support of the research funding by Lembaga Pengelola Dana Pendidikan (LPDP) through grant number SKPB-6667/LPDP/LPDP.3/2024.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] M. A. DeRousseau, J. R. Kasprzyk, and W. V. Srubar, “Computational Design Optimization of Concrete Mixtures: A Review,” Cement and Concrete Research, vol. 109, pp. 42–53, 2018.

[2] Guide to Evaluation of Strength Test Results of Concrete, ACI Standard 214R-11, 2011.

[3] Selecting Proportions for Normal-Density and High-Density Concrete Guide, ACI Standard PRC-211.1-22, 2022.

[4] A. I. Oviedo, J. M. Londoño, J. F. Vargas, C. Zuluaga, and A. Gómez, “Modeling and Optimization of Concrete Mixtures Using Machine Learning Estimators and Genetic Algorithms,” Modelling, vol. 5, no. 3, pp. 642-658, 2024.

[5] S. Zuo and B. Liu, “Optimization Design of Concrete Mix Proportion Based on Support Vector Machine Regression and Enhanced Genetic Algorithm,” Discover Applied Sciences, vol. 7, no. 3, article no. 200, 2025.

[6] A. A. Krimpenis and L. Athanasakos, “”Optimizing the Optimization”: A Hybrid Evolutionary-Based AI Scheme for Optimal Performance,” Computers, vol. 14, no. 3, article no. 97, 2025.

[7] B. F. Azevedo, A. M. A. C. Rocha, and A. I. Pereira, “Hybrid Approaches to Optimization and Machine Learning Methods: A Systematic Literature Review,” Machine Learning, vol. 113, no. 7, pp. 4055-4097, 2024.

[8] Y. Chen and J. Shi, “Multi-Objective Optimization for Complex Systems Considering Both Performance Optimality and Robustness,” Applied Sciences, vol. 14, no. 13, article no. 5371, 2024.

[9] E. Baş, “Binary Aquila Optimizer for 0–1 Knapsack Problems,” Engineering Applications of Artificial Intelligence, vol. 118, article no. 105592, 2023.

[10] M. Y. Cheng and M. N. Sholeh, “Artificial Satellite Search: A New Metaheuristic Algorithm for Optimizing Truss Structure Design and Project Scheduling,” Applied Mathematical Modelling, vol. 143, article no. 116008, 2025.

[11] K. Kiran, “A Benchmark Study on Steepest Descent and Conjugate Gradient Methods-Line Search Conditions Combinations in Unconstrained Optimization,” Croatian Operational Research Review, vol. 13, no. 1, pp. 77-97, 2022.

[12] S. Ruder, “An Overview of Gradient Descent Optimization Algorithms,” https://doi.org/10.48550/arXiv.1609.04747, 2017.

[13] S. Karimi and S. A. Vavasis, "Nonlinear Conjugate Gradient for Smooth Convex Functions," Mathematical Programming Computation, vol. 16, no. 2, pp. 229-254, 2024.
[14] G. B. Dantzig, "Linear Programming under Uncertainty," Management Science, vol. 1, pp. 197-206, 1955.
[15] S. I. Gass, "The First Linear-Programming Shoppe," Operations Research, vol. 50, no. 1, pp. 61-68, 2002.
[16] R. Bellman, I. Glicksberg, and O. Gross, "The Theory of Dynamic Programming as Applied to a Smoothing Problem," Journal of the Society for Industrial and Applied Mathematics, vol. 2, no. 2, pp. 82-88, 1954.
[17] R. Bellman, "The Theory of Dynamic Programming," Bulletin of the American Mathematical Society, vol. 60, no. 6, pp. 503-515, 1954.
[18] Y. Zhang, "A Survey of Dynamic Programming Algorithms," Applied and Computational Engineering, vol. 35, no. 1, pp. 183-189, 2024.
[19] F. Glover, "Future Paths for Integer Programming and Links to Artificial Intelligence," Computers and Operations Research, vol. 13, no. 5, pp. 533-549, 1986.
[20] S. Kirkpatrick, C. D. Jr. Gelatt, and M. P. Jr. Vecchi, "Optimization by Simulated Annealing," Science, vol. 220, no. 4598, pp. 671-680, 1983.
[21] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," Proceedings of the IEEE International Conference on Neural Networks (ICNN), pp. 1942-1948, 1995.
[22] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant System: Optimization by a Colony of Cooperating Agents," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 26, no. 1, pp. 29-41, 1996.
[23] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," Advances in Engineering Software, vol. 69, pp. 46-61, 2014.
[24] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197, 2002.
[25] C. A. C. Coello and M. S. Lechuga, "MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization," Proceedings of the 2002 Congress on Evolutionary Computation (CEC'02), pp. 1051-1056, 2002.
[26] S. Liu, H. Wang, W. Peng, and W. Yao, "Surrogate-Assisted Evolutionary Algorithms for Expensive Combinatorial Optimization: A Survey," Complex and Intelligent Systems, vol. 10, no. 3, pp. 5933-5949, 2024.
[27] R. K. Tipu, P. Rathi, K. S. Pandya, and V. R. Panchal, "Optimizing Sustainable Blended Concrete Mixes Using Deep Learning and Multi-Objective Optimization," Scientific Reports, vol. 15, no. 1, article no. 16356, 2025.
[28] I. C. Yeh, "Modeling of Strength of High-Performance Concrete Using Artificial Neural Networks," Cement and Concrete Research, vol. 28, no. 12, pp. 1797-1808, 1998.
[29] B. Abidhan, A. K. Alzo'ubi, S. Palanivelu, P. Hamidian, A. GuhaRay, G. Kumar, et al., "A Hybrid Approach of ANN and Improved PSO for Estimating Soaked CBR of Subgrade Soils of Heavy-Haul Railway Corridor," International Journal of Pavement Engineering, vol. 24, no. 1, article no. 2176494, 2023.

Copyright© by the authors. Licensee TAETI, Taiwan. This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-NC) license (https://creativecommons.org/licenses/by-nc/4.0/).