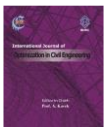
INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2025; 15(2):181-201



A TWO-PHASE METAMODEL-DRIVEN APPROACH FOR TOPOLOGY AND SIZE OPTIMIZATION OF TRUSS STRUCTURES

M. Ilchi Ghazaan[*,†] and M. Sharifi
School of Civil Engineering, Iran University of Science and Technology, P.O. Box 16846-13114, Iran

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

ABSTRACT

This paper introduces a novel two-phase metamodel-driven methodology for the simultaneous topology and size optimization of truss structures. The approach addresses critical limitations in computational efficiency and solution quality. The framework integrates the Flexible Stochastic Gradient Optimizer (FSGO) with adaptive sampling and machine learning to minimize the number of structural analyses (NSAs), while achieving lighter, high-performance designs. In Phase One, FSGO employs a dual global-local search strategy governed by Extensive Constraints (EC), a dynamic constraint relaxation mechanism to balance exploration of unconventional topologies and exploitation of optimal member sizes. By creating adaptive margins around design constraints, EC enables broader exploration of the design space while ensuring feasibility. Phase Two focuses on precision size optimization, leveraging pruned metamodels trained on critical regions of the design space to refine cross-sectional areas for the finalized topology. Comparative evaluations on benchmark planar and spatial trusses demonstrate the method's superiority: it reduces NSAs by 22–79% compared to state-of-the-art approaches and achieves 0.04–0.7% lighter designs while eliminating up to 31% of redundant members. Results validate the framework as a paradigm shift in truss optimization, merging computational efficiency with structural innovation.

Keywords: Machine learning, Metamodel, Adaptive sampling, Topology optimization, Size optimization.

Received: 10 March 2025; Accepted: 15 May 2025

[*]Corresponding author: School of Civil Engineering, Iran University of Science and Technology, P.O. Box 16846-13114, Iran
[†]E-mail address: ilchi@iust.ac.ir (M. Ilchi Ghazaan)

1. INTRODUCTION

Truss structural optimization is a complex engineering challenge requiring a balance between weight efficiency, load-bearing capacity, and constructability. Traditional methods often converge to suboptimal solutions due to the combinatorial complexity of topology and sizing decisions, limiting design innovation. To address these issues, researchers have proposed advanced techniques. Assimi et al. [1] introduced a multi-objective genetic programming approach with an adaptive mutant operator for simultaneous sizing and topology optimization, effectively identifying redundant members and generating trade-off solutions that satisfy multiple constraints such as stability, stress limits, and displacement. He et al. [2] developed a double-layer optimization framework incorporating local buckling stability into truss design, using linearized constraints to ensure efficiency and address practical concerns like transient hinges and section compatibility. Kaveh et al. [3] proposed a hybrid growth optimizer combined with an improved arithmetic optimization algorithm (IHGO) for discrete structural problems, enhancing exploration, exploitation, and solution retention, outperforming other metaheuristics in skeletal structure optimization tests.

Metaheuristic algorithms have revolutionized structural optimization by efficiently navigating complex, non-convex design spaces through stochastic search, enabling discovery of unconventional solutions missed by traditional methods. However, their practical use is often limited by high computational costs, especially in large-scale structures where each analysis is expensive. As problem complexity increases, so does the computational burden, hindering industrial adoption. Recent studies aim to address this challenge. Nemati et al. [4] introduced the Connected Banking System Optimization (CBSO) algorithm for truss sizing, demonstrating strong performance across six benchmark problems. Zhou et al. [5] enhanced the sine-cosine algorithm (SCA) with Lévy flight and elite guidance strategies, improving efficiency in size, shape, and topology optimization. Truong et al. [6] integrated the Rao algorithm with an improved k-nearest neighbor model (k-NNC), significantly reducing computational cost without sacrificing solution quality. Manguri et al. [7] reviewed recent advances in metaheuristics combined with machine learning, highlighting pathways to more efficient civil engineering design. Kaveh et al. [8] proposed a chaotic Water Strider Algorithm (WSA) using Circle map-based chaos, outperforming conventional methods in optimizing large trusses under frequency constraints, achieving lighter designs and faster convergence.

Recent advances in machine learning and surrogate modeling offer promising ways to reduce computational costs in structural optimization. Nourbakhsh et al. [9] introduced Generalizable Surrogate Models (GSMs) that use structural feature descriptors to accurately predict stress in 3D trusses across varying geometries, topologies, and boundary conditions, without retraining. Liu et al. [10] proposed a framework combining modular encoding, graph theory, and Radial Basis Function Neural Networks (RBFNN) to optimize multi-morphology lattice structures, showing how tailored microstructures can enhance mechanical performance. Song et al. [11] developed a Polynomial Chaos Kriging-based method for robust design optimization, integrating deterministic and random variables within an active-learning framework to boost efficiency and accuracy. Ren et al. [12] reviewed Differential Evolution (DE) algorithms for expensive optimization problems, highlighting improvements in framework design, surrogate-assisted methods, and parallel

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

computing, while identifying ongoing challenges. Negrin et al. [13] provided a comprehensive overview of metamodel-assisted structural design optimization (MASDO), offering practical insights into integrating these models into engineering workflows. Kaveh [14] explored the broader application of artificial neural networks in civil engineering, showcasing their effectiveness in structural optimization, material science, and predictive analysis through real-world examples like dome design and displacement forecasting. Despite these advancements, key challenges remain particularly in integrating data-driven models with optimization frameworks. Handling mixed discrete-continuous variables, managing conflicting constraints, and ensuring consistent solution accuracy across diverse structural configurations continue to pose significant hurdles.

Structural optimization continues to face fundamental challenges in balancing exploration and refinement during the design process. Sequential methods that separate topology generation from size optimization often miss critical interdependencies between these aspects, limiting overall performance. Effective constraint handling also remains a major issue: overly strict constraints stifle innovation, while loose enforcement leads to impractical designs. These difficulties become even more pronounced when addressing real-world factors such as dynamic loads, buckling resistance, and manufacturing constraints. Recent studies have introduced innovative approaches to tackle these challenges. Gao et al. [15] proposed an adaptive Gaussian Process Regression (GPR) framework for seismic collapse reliability prediction, combining adaptive sampling with a two-step global-local optimization. This method significantly improves accuracy while reducing computational costs compared to traditional Monte Carlo simulations. Megahed [16] applied adaptive sampling and machine learning to predict axial compression capacity in ECC-CES columns, generating interpretable design equations via symbolic regression. Peng et al. [17] introduced AK-SEUR, an adaptive Kriging-based learning function that reduces uncertainty in reliability analysis, showing strong performance on nonlinear problems with complex failure modes and low failure probabilities. Nath et al. [18] provided a comprehensive review of machine learning and deep learning applications in finite element analysis (FEA), emphasizing their potential to cut computational costs, reduce modeling time, and lessen reliance on expert input while highlighting opportunities for deeper integration of ML with FEA tools.

Building on recent advances, this paper introduces a novel two-phase optimization framework that combines the Flexible Stochastic Gradient Optimizer (FSGO) with metamodel-driven optimization and adaptive sampling. The method addresses key limitations of traditional approaches by integrating stochastic exploration and gradient-based refinement, significantly improving computational efficiency. Leveraging insights from multi-fidelity optimization, it incorporates machine learning and adaptive sampling to enhance performance across diverse truss design problems.

The remainder of this paper is structured to systematically address these challenges. Section 2 defines the truss optimization problem, presenting mathematical formulations for design variables, objectives, and constraints. Section 3 outlines the theoretical framework of the proposed optimization methodology, highlighting its key components and innovations. Section 4 details the implementation framework, including metamodel training and adaptive sampling strategies. Section 5 presents comprehensive validation studies on benchmark truss

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

structures, comparing the proposed method's performance against existing approaches. Finally, Section 6 concludes the paper with a discussion of practical implications.

2. PROBLEM DEFINITION

The optimization of truss structures involves a complex interplay of design variables, performance metrics, and constraints that must be carefully balanced to achieve both structural efficiency and feasibility. This section outlines the key components of the optimization problem, including the design variables, objective function, and constraints.

2.1 Design variables

The design variables in truss optimization can be categorized into two main types:

- **Topology Variables:** These variables determine the configuration of the truss structure, specifically which members are included or excluded from the design. The goal is to identify an optimal arrangement of these members that maximizes structural strength while minimizing weight.
- **Size Variables:** These represent the cross-sectional areas of the active truss members. The sizes must be optimized to ensure that the truss can withstand applied loads while maintaining structural integrity and satisfying all performance constraints.

2.2 Objective function

The primary objective of the optimization process is to minimize the overall weight of the truss structure while ensuring compliance with all design constraints. The objective function is mathematically expressed as:

$$\text{Minimize } W = \sum_{i=1}^{n} A_i l_i \rho \tag{1}$$

where W represents the total weight of the truss structure, A_i denotes the cross-sectional area of the ith truss element, l_i indicates the length of the ith truss element, ρ is the material density, and n is the total number of active truss elements.

2.2 Constraints

To ensure the design is both feasible and functional, the optimization process is subject to several constraints. These constraints are critical for maintaining structural integrity and meeting performance requirements. The key constraints are as follows:

- **Stress Constraints:** The stress in each truss element must not exceed the allowable material stress. This ensures that no member fails under the applied loads. Mathematically, this is expressed as:

$$\sigma_i \leq \sigma_{allowable} \qquad \text{for all } i \text{ in elements} \tag{2}$$

Where σ_i is the stress in the ith truss element, and $\sigma_{allowable}$ is the maximum allowable stress for the material.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

- **Displacement Constraints:** The displacements at critical nodes must remain within acceptable limits to prevent excessive deformation. This is particularly important for maintaining the functionality and stability of the structure. The constraint is given by:

$$\delta_j \leq \delta_{\max} \qquad \text{for all } j \text{ in nodes} \tag{3}$$

Where δ_j is the displacement at node j, and δ_{max} is the maximum allowable displacement.

- **Frequency Constraints:** To avoid resonance and ensure dynamic stability, the natural frequencies of the truss structure must satisfy specific bounds. This is especially critical for structures subjected to dynamic loads. The constraint is expressed as:

$$f_k \geq f_{\min} \qquad \text{for all } k \text{ in modes} \tag{4}$$

Where f_k is the kth natural frequency of the structure, and f_{min} is the minimum allowable frequency.

- **Geometric Constraints:** The cross-sectional areas of the truss elements must adhere to specified minimum and maximum limits. This ensures that the design remains practical and manufacturable. The constraint is expressed as:

$$A_{\min} \leq A_i \leq A_{\max} \qquad \text{for all } i \text{ in elements} \tag{5}$$

3. FLEXIBLE STOCHASTIC GRADIENT OPTIMIZER (FSGO)

The Flexible Stochastic Gradient Optimizer (FSGO) is a hybrid algorithm designed for global and local search in truss optimization. It balances exploration and exploitation by combining stochastic perturbations with gradient-based refinement. FSGO uses a metamodel to approximate the objective function, significantly reducing computational cost. The algorithm enhances search performance by dynamically adjusting step sizes and directions, enabling efficient navigation of complex design spaces. As shown in Figure 1, FSGO outperforms Gradient-Based Optimization (GBO) by exploring broader regions while maintaining precision in local refinement. This combination of stochastic and directed search makes FSGO highly effective for handling the coupled topology and sizing variables in truss design.

FSGO operates iteratively using distinct strategies for global and local searches. In global mode, it initializes a population via Latin Hypercube Sampling (LHS) across the design bounds (B) and selects the best candidate based on the metamodel ($\hat{G}$). For local searches, it starts from a user-defined point. During each iteration, FSGO generates new solutions by perturbing the current best solution (Z*) using step sizes from a predefined set (Γ). These steps follow random ascent or descent directions, scaled by the gradient magnitude of the metamodel to guide exploration. Solutions remain within $[L_n, U_n]$ to ensure feasibility. The metamodel evaluates each candidate, and the best one updates Z*. This continues for K/M loops (K: total iterations, M: population size), until convergence yields the optimal solution (Z*).

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

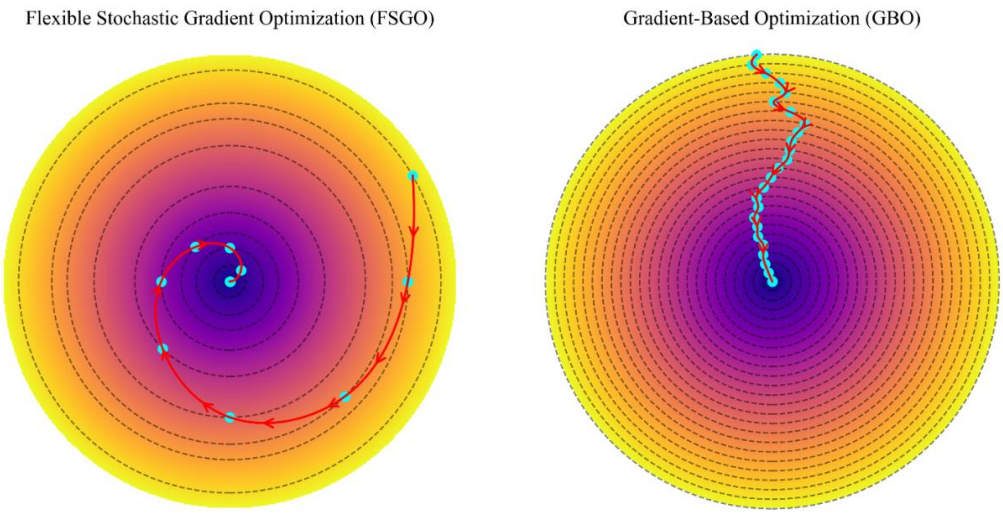


Figure 1: Comparison of Search Trajectories: Flexible Stochastic Gradient Optimization (FSGO) vs. Gradient-Based Optimization (GBO)

The strength of FSGO lies in its flexibility and efficiency, enabled by a metamodel that reduces computational overhead. Pseudocode 1 outlines the algorithm, highlighting its integration of stochastic exploration and directed search strategies. This combination ensures thorough exploration of the design space and accurate refinement of high-potential solutions.

Pseudocode 1: Flexible Stochastic Gradient Optimizer (FSGO)

Input:
- $N \in \mathbb{N}$: Number of design variables
- $B = [L_1, U_1] \times \ldots \times [L_n, U_n] \subset \mathbb{R}^N$: Design variable bounds
- $M \in \mathbb{N}$: Population size
- $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_m\} \subset \mathbb{R}^+$: Set of learning rates
- $\Omega \in$ {'GLOBAL', 'LOCAL'}: Search type
- $Z_0 \in \mathbb{R}^N$: Initial point (if Ω = 'LOCAL')
- $\hat{G}: \mathbb{R}^N \to \mathbb{R}$: Metamodel approximating the objective function

Output:
- $Z^* \in \mathbb{R}^N$: Optimal solution

Steps:

1. Initialization:
   if Ω = 'GLOBAL': // Initialize global search
      $Q \leftarrow \mathrm{LHS}(B, M)$ // Generate initial population using Latin Hypercube Sampling
      $Z^* \leftarrow \mathrm{argmin}_\{Z \in Q\}\ \hat{G}(Z)$ // Find best solution in initial population
   else if Ω = 'LOCAL': // Initialize local search
      $Z^* \leftarrow Z_0$ // Use provided initial point
2. Main Loop:
   $R \leftarrow \lfloor K / M \rfloor$ // Compute number of loops
   for r = 1 to R do:
      $y^* \leftarrow \nabla\hat{G}(Z^*)$ // Compute gradient of metamodel at current solution

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]
[DOI: 10.22068/ijoce.2025.15.2.630]

```
    Q_new ← {}  // Initialize new population
    for m = 1 to M do:
      W ← {}  // Create new candidate solution
      for n = 1 to N do:
        γ ← Random(Γ)  // Select random learning rate
        ΔD ← Random({'ASCENT', 'DESCENT'})  // Randomly choose direction
        if ΔD = 'DESCENT':
          Δ ← -γ · |y*[n]|  // Step proportional to gradient magnitude (descent)
        else if ΔD = 'ASCENT':
          Δ ← γ · |y*[n]|  // Step proportional to gradient magnitude (ascent)
        W[n] ← Z*[n] + Δ  // Update candidate variable
        W[n] ← max(L_n, min(U_n, W[n]))  // Enforce bounds
      Add W to Q_new
    For each W ∈ Q_new:
      f(W) ← Ĝ(W)  // Evaluate fitness using metamodel
    W* ← argmin_{W ∈ Q_new} f(W)  // Identify best candidate
    Z* ← W*  // Update current best solution
  3. Return:
    - Z*  // Return optimal solution
```

4. METHODOLOGY

This section introduces an advanced two-phase optimization framework that enhances truss design by integrating metamodeling with adaptive sampling. The methodology addresses high computational costs and local optima issues through a systematic decomposition approach. Phase One simultaneously explores optimal topologies and refines member sizes using the Flexible Stochastic Gradient Optimizer (FSGO). Phase Two focuses on precision size optimization for the finalized topology. Key innovations include dynamic constraint handling via Extensive Constraints (EC), adaptive metamodeling to improve prediction accuracy in critical regions, and efficient resource allocation that minimizes redundant structural analyses while maintaining solution quality.

The framework achieves computational efficiency through synergistic mechanisms. Latin Hypercube Sampling (LHS) provides a robust initial dataset, ensuring broad design space coverage. FSGO's hybrid stochastic-gradient search balances exploration of unconventional designs with exploitation of promising solutions. Adaptive sampling allocates resources strategically, focusing on critical regions. By integrating machine learning with structural optimization, the methodology systematically decomposes topology and size variables, effectively managing high dimensionality.

Figure 2 presents an overview of the two-phase optimization methodology, highlighting its sequential yet integrated structure. Phase One performs simultaneous topology and size optimization using FSGO, guided by Extensive Constraints. Phase Two refines the resulting topology through precise size optimization, employing pruned metamodels to deliver lightweight, high-performance designs.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

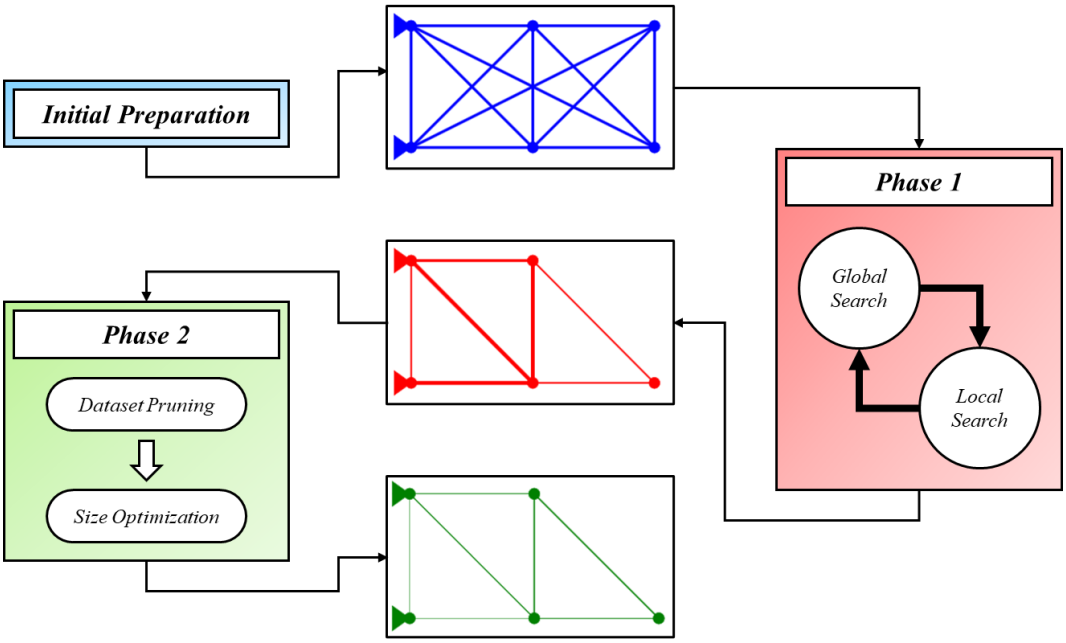


Figure 2: Overview of the Two-Phase Optimization Methodology

4.1 Initial Preparation

The optimization process begins by building a robust foundation for the metamodel, which serves as the framework's core. Latin Hypercube Sampling (LHS) generates an initial set of design configurations that span the parameter space. Each configuration undergoes finite element analysis (FEA) to assess structural responses, such as stress, displacement, and natural frequency, which together form the training dataset. An XGBoost metamodel is then trained on this data, chosen for its accuracy and efficiency in handling high-dimensional problems. By approximating structural behavior, the metamodel minimizes reliance on costly FEA evaluations during optimization, accelerating convergence to optimal solutions.

Pseudocode 2 details the initial preparation phase, which systematically generates a dataset and trains the metamodel. This phase establishes the foundation for subsequent optimization stages.

Pseudocode 2: Initial Preparation

Input:
- $N \in \mathbb{N}$: Number of design variables
- $B = [L_1, U_1] \times \ldots \times [L_n, U_n] \subset \mathbb{R}^N$: Design variable bounds
- $M \in \mathbb{N}$: Population size

Output:
- $\mathcal{D}$: Dataset of samples and responses
- $\hat{G}$: Trained metamodel

Steps:
1. $X \leftarrow \mathrm{LHS}(B, M)$ // Generate initial samples using Latin Hypercube Sampling

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]
[DOI: 10.22068/ijoce.2025.15.2.630]

2. For each $x_i \in X$:
   $r_i \leftarrow \text{FEA}(x_i)$ // Evaluate structural response using Finite Element Analysis
3. $\mathcal{D} \leftarrow \{(X, R)\}$ // Construct dataset from samples and responses
4. $\hat{G} \leftarrow \text{TrainMetamodel}(\mathcal{D})$ // Train surrogate model on dataset
5. Return:
   - $\mathcal{D}$ // Return dataset
   - $\hat{G}$ // Return trained metamodel

4.2 Phase One: Topology and Size Optimization

The first optimization phase focuses on simultaneously determining the optimal topology and member sizes by leveraging the Flexible Stochastic Gradient Optimizer (FSGO) to navigate the complex design space. FSGO balances global exploration with local refinement, starting with a broad search and gradually focusing on promising regions. A key innovation is the use of Extensive Constraints (EC), a dynamic constraint relaxation mechanism that temporarily loosens performance criteria to encourage exploration of unconventional, high-performance designs. As optimization progresses, EC systematically tightens constraints to ensure final designs meet engineering requirements. Throughout this phase, the metamodel is continuously refined using new simulation results, improving prediction accuracy in critical areas, enabling the efficient identification of lightweight, structurally sound configurations while maintaining computational efficiency.

Figure 3 illustrates the Extensive Constraints (EC) mechanism, which dynamically adjusts constraint boundaries during optimization. It shows three scenarios: (a) when a value must be less than C, (b) when it must be greater than C, and (c) when it must equal C. This adaptive approach expands design space exploration while maintaining feasibility. Pseudocode 3 outlines Phase One implementation, demonstrating how FSGO, EC, and adaptive sampling are integrated to guide the search effectively.

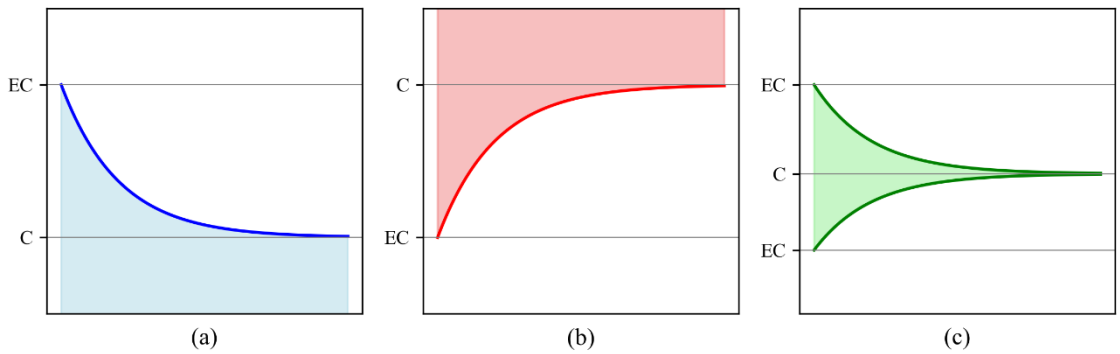


Figure 3: Representation of Extensive Constraints (EC): a) when a lesser value than C needs to be satisfied, b) when a greater value than C needs to be satisfied, and c) when a value equal to C needs to be satisfied.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

Pseudocode 3: Phase I

```
Input:
  - 𝒟: Dataset of samples and responses
  - Ĝ: Trained metamodel
  - B = [L₁, U₁] × ... × [Lₙ, Uₙ] ⊂ ℝᴺ: Design variable bounds
  - RC ∈ ℝᵏ: Real constraints
  - Γ = {γ₁, γ₂, ..., γₘ} ⊂ ℝ⁺: Set of learning rates
  - Ω ∈ {'GLOBAL', 'LOCAL'}: Search type
  - α ∈ ℝ⁺: Decay rate for Acceptable Margin (AM)
Output:
  - Z*: Optimized design
  - 𝒟: Updated dataset
  - S: Set of inactive variables
Steps:
1. Initialization:
  Z* ← arg min_{x∈𝒟} Ĝ(x)  // Select best initial design from dataset
  Ω ← 'GLOBAL'  // Start with global search mode
  i ← 0  // Iteration counter
2. Optimization Loop:
  while ∃j: |EC_j(Z*) - RC_j| > 1×10⁻⁶:  // Continue until Extensive Constraints (EC) are satisfied
      AM ← 0.1·e^(-α·i)  // Compute Acceptable Margin (AM) using exponential decay
      For each C_i ∈ RC:  // Adjust constraints based on AM
        If C_i ≥ C'_i:
          EC_i ← C'_i · (1 − AM)  // Relax upper bound
        Else if C_i ≤ C'_i:
          EC_i ← C'_i · (1 + AM)  // Relax lower bound
        Else:
          EC_i ← C'_i · (1 ± AM)  // Adjust equality constraint
      if Ω = 'GLOBAL':
        W* ← FSGO_GLOBAL(Ĝ, EC, B)  // Perform global search
      else:
        W* ← FSGO_LOCAL(Ĝ, Z*, EC, B)  // Perform local search
      r_w ← FEA(W*)  // Evaluate structural response of candidate
      𝒟 ← 𝒟 ∪ {(W*, r_w)}  // Update dataset
      Retrain Ĝ on 𝒟  // Retrain metamodel
      if ℱ(W*) < ℱ(Z*):  // Check for improvement in objective function
        Z* ← W*  // Update best solution
        improved ← True
      else:
        improved ← False
      if not improved:  // Switch search mode if no improvement
        Ω ← 'LOCAL' if Ω = 'GLOBAL' else 'GLOBAL'
      i ← i + 1  // Increment iteration counter
```

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

3. Identify Inactive Variables:
   $S \leftarrow \{j \mid Z^*_j = 0\}$ // Identify inactive variables
4. Return:
   - Z^* // Return optimized design
   - $\mathcal{D}$ // Return updated dataset
   - S // Return set of inactive variables

4.3 Phase Two: Size Optimization

Building upon the optimal topology from Phase One, this final phase focuses on refining member cross-sectional areas to achieve a lightweight yet robust design. The problem is simplified by removing inactive design variables, narrowing the optimization space. The metamodel is retrained on this reduced set to improve predictive accuracy for the current topology. FSGO then performs gradient-based local search to precisely optimize member sizes, identifying the lightest configuration that satisfies all constraints. The process iterates until improvements diminish, ensuring convergence to a high-quality solution. This phase highlights how targeted refinement enhances performance while preserving computational efficiency.

Pseudocode 4 details Phase Two, focusing on retraining the metamodel using the pruned dataset and iteratively refining member sizes. This phase demonstrates how the two-phase decomposition strategy enhances both computational efficiency and solution quality.

Pseudocode 4: Phase II

Input:
   - Z^*: Optimized design from Phase I
   - $\mathcal{D}$: Dataset from Phase I
   - S: Set of inactive variables from Phase I
   - $\hat{G}$: Trained metamodel
   - $B = [L_1, U_1] \times \ldots \times [L_n, U_n] \subset \mathbb{R}^N$: Design variable bounds
   - $RC \in \mathbb{R}^k$: Real constraints
   - $\varepsilon \in \mathbb{R}^+$: Critical area threshold
Output:
   - Z^*: Final optimized design
Steps:
1. Update Bounds for Active Variables:
   $B' \leftarrow [L_1', U_1'] \times \ldots \times [L_\{n\text{-}|S|\}', U_\{n\text{-}|S|\}']$ // Update bounds for active variables
   For each active variable $j \notin S$:
      $L_j' \leftarrow \max(\varepsilon, L_j)$ // Apply critical area threshold
      $U_j' \leftarrow U_j$ // Preserve original upper bound
   $\mathcal{D}' \leftarrow \{(x', r) \mid (x, r) \in \mathcal{D}, x' = (x_j \mid j \notin S)\}$ // Prune dataset
2. Retrain Metamodel:
   $\hat{G}' \leftarrow \text{TrainMetamodel}(\mathcal{D}')$ // Retrain metamodel on pruned dataset
3. Extract Active Components:
   $Y^* \leftarrow (Z^*_j \mid j \notin S)$ // Extract active components of solution

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

```
    W_prev ← Weight(FEA(Y*))  // Compute initial weight
  4. Refine Design Iteratively:
    repeat:  // Refine design iteratively
      Y ← FSGO_LOCAL(Ĝ', Y*, RC, B')  // Perform local search
      W_curr ← Weight(FEA(Y))  // Compute weight of new design
      ΔW ← |(W_prev - W_curr)/W_prev|  // Compute relative weight change
      if ΔW < 1×10^-4:  // Check for stagnation
        stagnation ← stagnation + 1
      else:
        stagnation ← 0
        W_prev ← W_curr
        Y* ← Y
    until stagnation = 100  // Terminate after 100 consecutive stagnations
  5. Reconstruct Full-Dimensional Solution:
    Z* ← Reconstruct(Y*, S)  // Reconstruct full-dimensional solution
  6. Return:
     - Z*  // Return final optimized design
```

5. NUMERICAL EXAMPLES

To validate the proposed two-phase optimization framework, we tested it on three increasingly complex truss structures: a 24-bar planar truss, a 39-bar planar truss with grouped elements, and a 72-bar spatial truss with multiple loading conditions. These benchmarks were selected to evaluate the method's performance across a range of structural types, from simple planar systems to complex 3D geometries with non-structural masses. Each case study assesses the algorithm's ability to perform simultaneous topology and size optimization under strict computational efficiency constraints. Initial and optimized configurations for each structure are shown in Figures 4, 6, and 8, clearly demonstrating the topology simplification achieved.

5.1 24-Bar Truss

The 24-bar planar truss comprises 24 elements connecting 8 nodes, with 6 essential nodes fixed for structural stability. Cross-sectional areas vary continuously between 1 cm² and 40 cm², where values below 1 cm² indicate element removal. The material has a Young's modulus of 69 GPa and a density of 2740 kg/m³. Each node carries a 5 kg mass, with an additional 500 kg non-structural mass at node 3. The design must satisfy displacement limits of 10 mm in the y-direction at nodes 5 and 6, stress limits of 172.43 MPa in all members, and a minimum first natural frequency of 30 Hz. Figure 4 shows the initial and optimized configurations, highlighting the removal of redundant elements and resulting simplified topology. Further details on this benchmark can be found in [19-21].

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

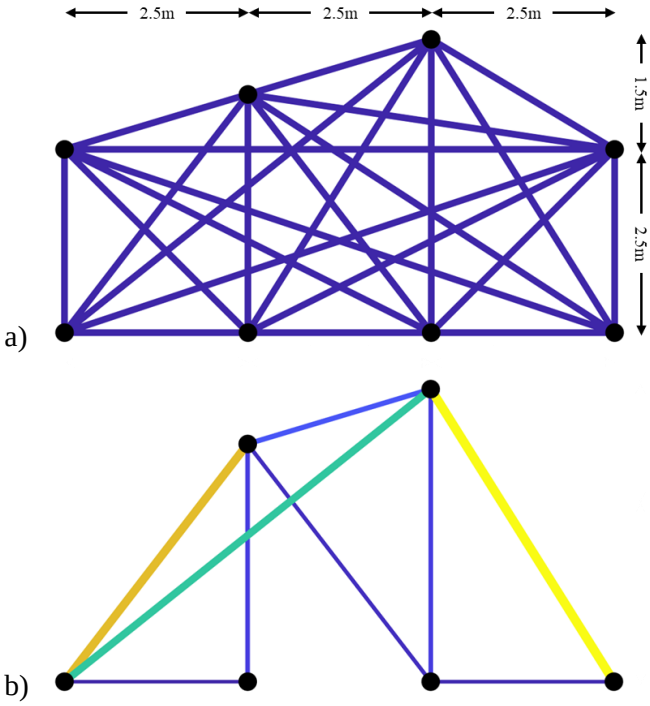


Figure 4: Comparison of the initial ground structure (a) and optimized size and topology (b) for a 24-bar truss structure

Table 1: Optimal parameters for the 24-bar truss with different methods.

Element no.	CSS [22]	PSO [22]	TLBO [23]	FTISA [24]	Proposed Methodology
A7	19.2000	20.1000	19.0214	19.0026	19.0026
A8	3.0000	14.8000	2.9134	2.8998	2.8998
A9	1.4000	Removed	1.7381	1.7757	1.7757
A12	4.0000	2.4000	4.5314	4.5443	4.5451
A13	14.1300	14.9000	13.8208	13.8095	13.8088
A14	Removed	1.2000	Removed	Removed	Removed
A15	3.3000	6.5000	2.8950	2.8539	2.8533
A16	23.9000	23.9000	23.8920	23.8830	23.8830
A22	Removed	Removed	Removed	1.0000	1.000
A23	1.0400	4.7000	1.1296	Removed	Removed
A24	1.4000	22.1000	1.3115	1.1893	1.1900
Best weight (kg)	119.75	151.63	119.1304	118.8996	118.8988
Mean weight (kg)	130.5	190.80	164.9053	122.0992	124.3729
Std. (kg)	5.44	22.16	32.1632	2.9234	3.5847
NSAs	2,320 [24]	2,380 [24]	20,000	1,950	1,524

Table 1 presents the optimal parameters for the 24-bar truss using various optimization methods. The proposed method achieved an optimal weight of 118.90 kg, a

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

0.7% reduction compared to FTISA (119.13 kg), while eliminating four structural elements (A_9, A_{14}, A_{22}, A_{23}). It also demonstrated high computational efficiency, requiring only 1,524 NSAs, a 22% reduction from FTISA's 1,950 NSAs. Across 30 independent runs, the method showed superior consistency, with a standard deviation of 3.58 kg versus TLBO's 32.16 kg, confirming the metamodel's stability. The optimization converged in 18 generations, with 85% of the effort focused on Phase One for topology exploration. Figure 5 illustrates the convergence profile of the penalized weight, showing a steady reduction throughout the optimization process.

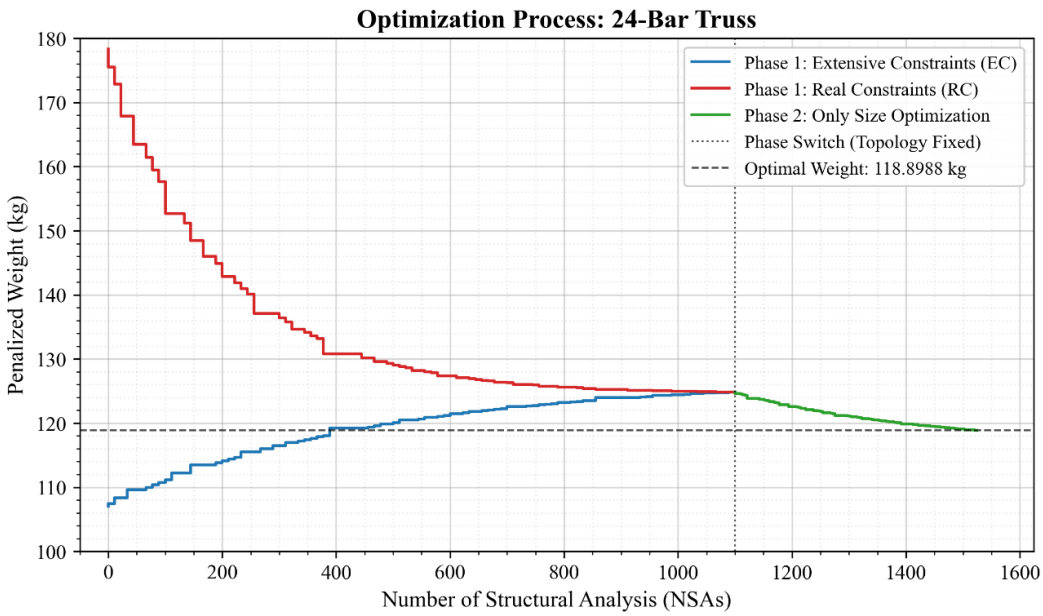


Figure 5: Convergence Profile of Penalized Weight During Optimization Process for the 24-Bar Truss

5.2 39-Bar Truss

The 39-bar planar truss comprises 39 elements connected by 12 nodes, with 5 essential nodes fixed for stability. The 21 element groups have cross-sectional areas ranging from –2.25 in² to 2.25 in², where values below 0.05 in² indicate member removal. Material properties include a Young's modulus of 10,000 ksi and a density of 0.1 lb/in³. All members must withstand stress below 20 ksi in tension and compression, and nodal displacements are limited to 2 inches. The design also satisfies fundamental frequency constraints. Figure 6 compares the initial and optimized configurations, showing significant topology simplification. This problem has been widely studied, including in [25] and [26].

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

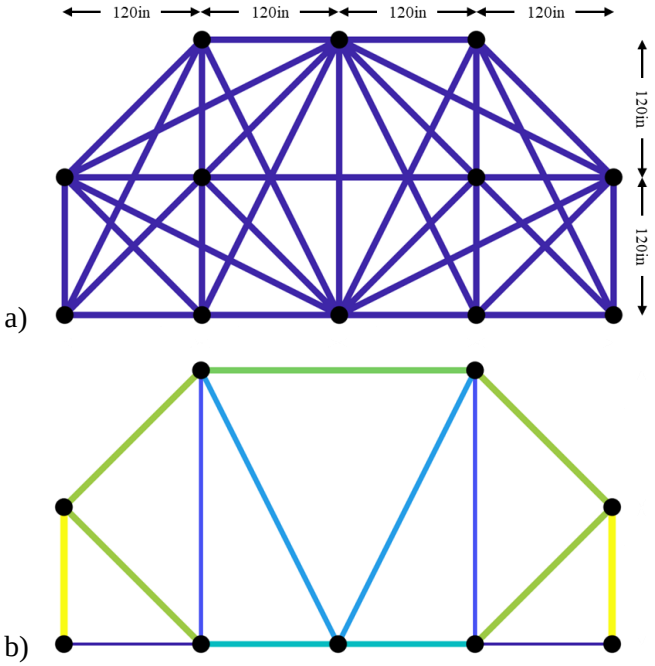


Figure 6: Comparison of the initial ground structure (a) and optimized size and topology (b) for a 39-Bar Truss Structure

Table 2: Optimal parameters for the 39-bar truss with different methods.

Element group no.	GA [27]	FA [28]	SOGP [29]	FTISA [24]	Proposed Methodology
G1	Removed	0.0500	0.0510	0.0500	0.0500
G2	0.7510	0.7524	0.7500	0.7500	0.7500
G3	0.0510	Removed	Removed	Removed	Removed
G5	1.5020	1.5001	1.5010	1.5000	1.500
G7	0.0520	Removed	Removed	Removed	Removed
G8	0.2510	0.2504	0.2500	0.2500	0.2500
G9	0.0510	Removed	Removed	Removed	Removed
G10	1.0610	1.0647	1.0610	1.0607	1.0607
G11	1.0630	1.0612	1.0610	1.0607	1.0607
G14	0.5590	0.5604	0.5595	0.5591	0.5591
G21	1.0050	1.0016	Removed	1.0002	1.0000
Best weight (kg)	196.546	193.5472	193.29	193.211	193.2072
Mean weight (kg)	-	207.34	-	195.1678	197.8154
Std. (kg)	-	10.2580	-	1.1193	1.4926
NSAs	-	50,000	-	10,100	2,127

Table 2 presents the optimal results for the 39-bar truss using various methods. The proposed approach achieved a minimum weight of 193.21 kg, outperforming FTISA by 0.04% (193.21 kg vs. 193.29 kg), while eliminating three element groups (G_3, G_7, G_9). It

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

demonstrated high computational efficiency, requiring only 2,127 NSAs, a 79% reduction compared to FTISA's 10,100 NSAs. Across 50 runs, the coefficient of variation was just 0.12%, much lower than FA's 5.3%, confirming the metamodel's stability. Phase Two accounted for only 31% of total NSAs, highlighting the effectiveness of the two-phase decomposition strategy in handling grouped elements. Figure 7 shows the convergence profile of penalized weight, illustrating the method's steady improvement during optimization.

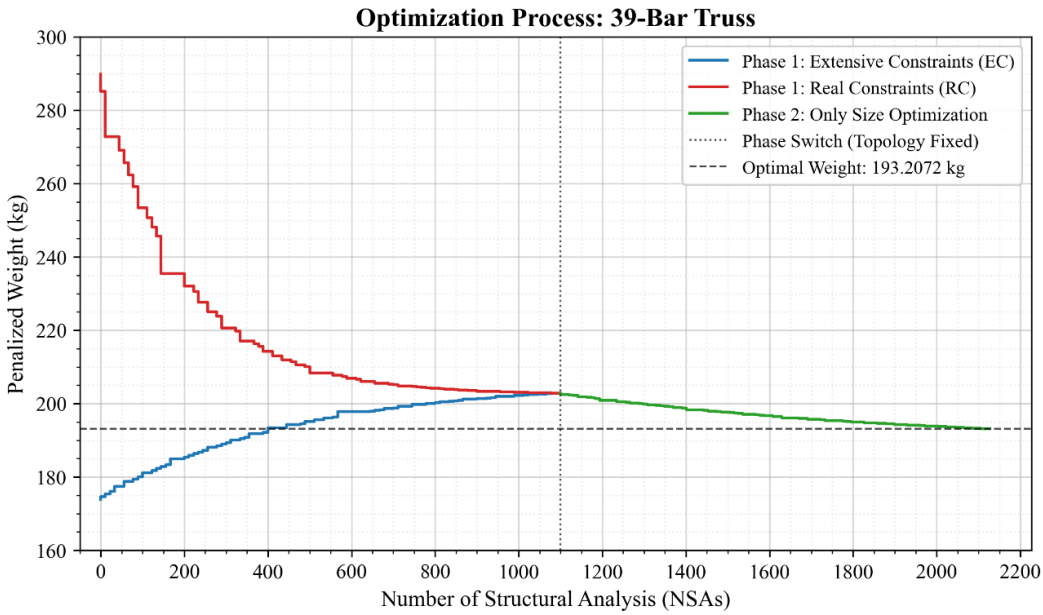


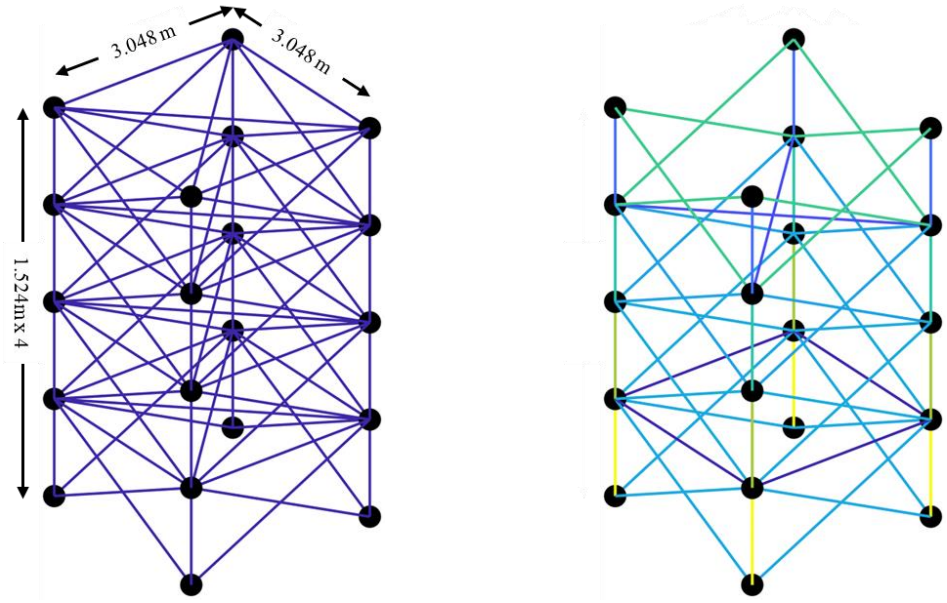
Figure 7: Convergence Profile of Penalized Weight During Optimization Process for the 39-Bar Truss

5.3 72-Bar Truss

The 72-bar spatial truss comprises 72 elements connected by 20 nodes, with 8 essential nodes fixed for structural stability. The structure includes 16 element groups with continuous cross-sectional areas ranging from 1 cm² to 30 cm², where values below 1 cm² indicate member removal. It uses a material with a Young's modulus of 68.95 GPa and density of 2767.99 kg/m³. Each node carries a 5 kg mass, and non-structural lumped masses of 2270 kg are applied at nodes 17–20. The design must satisfy displacement limits of 6.35 mm in x- and y-directions at top nodes, stress constraints under 172.375 MPa, and dual frequency requirements: a minimum first natural frequency of 4 Hz and third natural frequency of 6 Hz. Figure 8 presents the initial and optimized configurations, showing significant topology simplification achieved through the optimization. This benchmark has been widely studied, including in [30] and [31], which offer detailed formulations and comparative results.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]



(a) (b)

Figure 8: Comparison of the initial ground structure (a) and optimized size and topology (b) for a 72-bar truss structure

Table 3: Optimal parameters for the 72-bar truss with different methods.

Element group no.	CSS [22]	PSO [22]	TLBO [23]	FTISA [24]	Proposed Methodology
G1	20.3200	21.1700	15.1415	16.0581	15.9429
G2	7.9600	9.5600	7.5945	7.5718	7.5883
G3	Removed	Removed	1.8944	1.979	1.9563
G5	10.0100	22.5800	12.6451	11.9596	12.0807
G6	8.1500	6.9800	7.6935	7.6899	7.6724
G8	Removed	5.1100	Removed	Removed	Removed
G9	8.0700	5.1600	10.0541	9.4584	9.4681
G10	8.0400	9.4800	7.2706	7.2924	7.2944
G11	3.1300	Removed	Removed	Removed	Removed
G12	Removed	Removed	3.7589	3.8901	3.8877
G13	5.5400	5.3000	4.9192	5.0018	4.9875
G14	8.0600	6.9800	10.2376	10.2378	10.2345
G15	Removed	5.6000	Removed	Removed	Removed
G16	9.0400	13.5600	Removed	Removed	Removed
Best weight (kg)	437.8500	504.0600	435.4325	435.0786	434.9919
Mean weight (kg)	456.95	559.11	400.8125	438.0922	442.7031
Std. (kg)	3.16	27.15	58.4291	2.23	2.9175
NSAs	5,400 [24]	8,300 [24]	20,000	2,490	1,874

Table 3 presents the optimal results for the 72-bar truss obtained using various optimization methods. The proposed methodology achieved the lowest weight of 434.99

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

kg while eliminating five element groups (G_3, G_8, G_{11}, G_{15}, G_{16}). With only 1,874 structural analyses (NSAs), the method demonstrated superior computational efficiency, reducing the number of evaluations by 24.7% compared to FTISA's 2,490 NSAs. Across 25 independent runs, the optimization showed consistent convergence (standard deviation of 2.92 kg), with adaptive sampling directing 72% of evaluations to critical regions of the design space. The final design satisfied all structural constraints while reducing the number of active elements by 31% from the initial configuration. Figure 9 illustrates the convergence behavior, showing a steady decline in penalized weight throughout the optimization process.

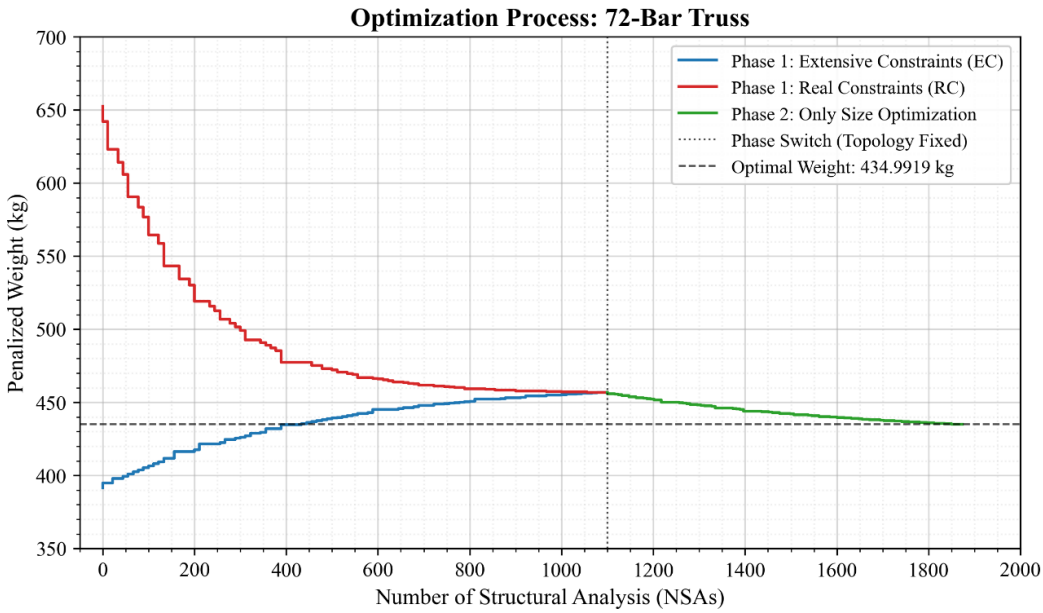


Figure 9: Convergence Profile of Penalized Weight During Optimization Process for the 72-Bar Truss

6. CONCLUSION

This study presents a two-phase optimization methodology that advances truss design by enhancing computational efficiency, solution quality, and the balance between exploration and exploitation. The framework integrates three key innovations: the Flexible Stochastic Gradient Optimizer (FSGO) for efficient global-local search, Extensive Constraints (EC) as a dynamic constraint handling mechanism, and a decomposition strategy that decouples topology and size optimization to boost performance.

The Flexible Stochastic Gradient Optimizer (FSGO) serves as the computational core of the proposed methodology, offering enhanced flexibility and efficiency. FSGO integrates stochastic gradient methods with population-based search strategies, enabling robust exploration of complex, non-convex design spaces. It operates in both bounded and point-initiated modes, adapting to various optimization scenarios. By leveraging metamodels to

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

approximate structural responses, FSGO significantly lowers computational cost while maintaining high accuracy. This hybrid approach ensures a balanced trade-off between exploration and exploitation, making it well-suited for large-scale, computationally intensive problems.

A key innovation is the phase-switching mechanism, which balances exploration and exploitation during optimization. In Phase One, the framework optimizes topology and member sizes by alternating between global and local search modes based on performance improvements. This process leverages the Extensive Constraints (EC) framework, which dynamically adjusts constraints to encourage broad exploration before enforcing strict feasibility. By adaptively relaxing constraints, EC uncovers unconventional, high-performance designs often missed by traditional methods, ensuring both flexibility and rigor in achieving superior structural solutions.

The proposed two-phase framework addresses the coupled challenges of topology and size optimization systematically. Phase One uses FSGO and EC to explore promising topologies, balancing exploration and exploitation. Phase Two refines member sizes using pruned metamodels trained on critical design regions. This strategy demonstrates strong performance across benchmark trusses (24-bar, 39-bar, and 72-bar), achieving superior designs with 22–79% fewer structural analyses and eliminating up to 31% of redundant members without compromising structural integrity or performance, showcasing its potential to transform truss optimization.

REFERENCES

1. Assimi H, Jamali A, Nariman-Zadeh N. Multi-objective sizing and topology optimization of truss structures using genetic programming based on a new adaptive mutant operator. *Neural Comput Appl*. 2019;**31**(10):5729-49.
2. He F, Feng R, Cai Q. Topology optimization of truss structures considering local buckling stability. *Comput Struct*. 2024;**294**:107273.
3. Kaveh A, Hamedani KB. A hybridization of growth optimizer and improved arithmetic optimization algorithm and its application to discrete structural optimization. *Comput Struct*. 2024;**303**:107496.
4. Nemati M, Zandi Y, Sabouri J. Truss sizing optimum design using a metaheuristic approach: Connected banking system. *Heliyon*. 2024;**10**(20).
5. Zhou H, et al. Improved sine-cosine algorithm for the optimization design of truss structures. *KSCE J Civ Eng*. 2024;**28**(2):687-98.
6. Truong VH, Pham HA, Tangaramvong S. An efficient method for nonlinear inelastic truss optimization based on improved k-nearest neighbor comparison and Rao algorithm. *Struct*. 2025.
7. Manguri A, et al. Topology, size, and shape optimization in civil engineering structures: A review. *CMES Comput Model Eng Sci*. 2025;**142**(2):933-71.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

8. Kaveh A, et al. Frequency-constrained optimization of large-scale dome-shaped trusses using chaotic water strider algorithm. *Struct*. 2021.
9. Nourbakhsh M, Irizarry J, Haymaker J. Generalizable surrogate model features to approximate stress in 3D trusses. *Eng Appl Artif Intell*. 2018;**71**:15-27.
10. Liu J, et al. A novel digital unit cell library generation framework for topology optimization of multi-morphology lattice structures. *Compos Struct*. 2025;**354**:118824.
11. Song C, et al. Analytical robust design optimization for hybrid design variables: An active-learning methodology based on polynomial chaos Kriging. *Reliab Eng Syst Saf*. 2024;**250**:110286.
12. Ren C, Meng Z. A survey on expensive optimization problems using differential evolution. *Appl Soft Comput*. 2025:112727.
13. Negrin I, Kripka M, Yepes V. Metamodel-assisted design optimization in the field of structural engineering: A literature review. *Struct*. 2023.
14. Kaveh A. Applications of artificial neural networks and machine learning in civil engineering. Springer; 2024.
15. Gao J, Du K, Lin J. A structural seismic reliability prediction method based on adaptive sampling and Gaussian process regression. *Struct*. 2025.
16. Megahed K. Prediction of the axial compression capacity of ECC-CES columns using adaptive sampling and machine learning techniques. *Sci Rep*. 2025;**15**(1):4181.
17. Peng C, et al. AK-SEUR: An adaptive Kriging-based learning function for structural reliability analysis through sample-based expected uncertainty reduction. *Struct Saf*. 2024;**106**:102384.
18. Nath D, et al. Application of machine learning and deep learning in finite element analysis: A comprehensive review. *Arch Comput Methods Eng*. 2024;**31**(5):2945-84.
19. Xu B, et al. Topology group concept for truss topology optimization with frequency constraints. *J Sound Vib*. 2003;**261**(5):911-25.
20. Gonçalves MS, Lopez RH, Miguel LFF. Search group algorithm: A new metaheuristic method for the optimization of truss structures. *Comput Struct*. 2015;**153**:165-84.
21. Kaveh A, Mahdavi V. Colliding bodies optimization for size and topology optimization of truss structures. *Struct Eng Mech*. 2015;**53**(5):847-65.
22. Kaveh A, Zolghadr A. Topology optimization of trusses considering static and dynamic constraints using the CSS. *Appl Soft Comput*. 2013;**13**(5):2727-34.
23. Savsani VJ, Tejani GG, Patel VK. Truss topology optimization with static and dynamic constraints using modified subpopulation teaching–learning-based optimization. *Eng Optim*. 2016;**48**(11):1990-2006.
24. Mortazavi A. A new fuzzy strategy for size and topology optimization of truss structures. *Appl Soft Comput*. 2020;**93**:106412.
25. Luh GC, Lin CY. Optimal design of truss structures using ant algorithm. *Struct Multidiscip Optim*. 2008;**36**(4):365-79.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]

26. Wu CY, Tseng KY. Truss structure optimization using adaptive multi-population differential evolution. *Struct Multidiscip Optim*. 2010;**42**(4):575-90.
27. Deb K, Gulati S. Design of truss-structures for minimum weight using genetic algorithms. *Finite Elem Anal Des*. 2001;**37**(5):447-65.
28. Miguel LFF, Lopez RH, Miguel LFF. Multimodal size, shape, and topology optimisation of truss structures using the Firefly algorithm. *Adv Eng Softw*. 2013;**56**:23-37.
29. Assimi H, Jamali A, Nariman-Zadeh N. Sizing and topology optimization of truss structures using genetic programming. *Swarm Evol Comput*. 2017;**37**:90-103.
30. Tejani GG, et al. Topology optimization of truss subjected to static and dynamic constraints by integrating simulated annealing into passing vehicle search algorithms. *Eng Comput*. 2019;**35**:499-517.
31. Savsani V, Tejani G, Patel V. Truss Optimization. Springer; 2024.

[Downloaded from cefsse.iust.ac.ir on 2026-10-20]

[DOI: 10.22068/ijoce.2025.15.2.630]