# Optimal design of structures using metaheuristics and metamodeling

Érica C.R. Carvalho[1], João Marcos P. Vieira[1], Patrícia H. Hallak[2], Afonso C.C. Lemonge[2]

[1]*Graduate Program of Computational Modeling, Federal University of Juiz de Fora*
*José Lourenço Kelmer St, São Pedro, Juiz de Fora, MG 36036-900, Brazil*
*ericacrcarvalho@gmail.com, joaomarcos.vieira@estudante.ufjf.br*
[2]*Dept. of Applied and Computational Mechanics, School of Engineering, Federal University of Juiz de Fora*
*José Lourenço Kelmer St, São Pedro, Juiz de Fora, MG 36036-900, Brazil*
*patricia.hallak@ufjf.br, afonso.lemonge@ufjf.br*

**Abstract.** Computation-intensive structural design optimization problems are common in engineering. The computation burden is often caused by expensive simulation models and becomes a problem in practice. To address such a challenge, metamodeling techniques are often chosen to improve the optimization efficiency. Metamodels, or surrogate models, are very popular methods and are considered a valuable tool to support a wide scope of activities in modern engineering design. The paper discusses strategies to obtain suitable metamodels coupled with a metaheuristic to assess their quality concerning prediction, considering machine learning techniques used in the literature. A multi-objective truss structural optimization problem is used to illustrate the approach's applicability, considering the structure's weight and critical load factor as objective functions concerning the truss structure's global stability. The numerical results demonstrate the efficiency and computational advantages of the proposed methodology.

**Keywords:** Structural optimization problems, Metamodels, Metaheuristics.

## 1 Introduction

In recent years, the application of metaheuristic techniques to solve multi-objective optimization problems has become an active field of research [1–4]. Metaheuristics are strategies that guide the search process and avoid the persistence of results in local maxima and minima, and they can use anything from simple local search procedures to complex learning processes [5]. The goal is to efficiently explore the search space to find optimal solutions or as close to them as possible.

However, the optimization process is usually very computationally intensive, as evaluating a single function can take hours or even days for extensive applications. For this reason, metamodeling is an efficient alternative to reduce the computational costs in the optimization process while maintaining the quality of the solutions. These models try to avoid using computationally expensive evaluations to obtain (i) better results with the same processing time or (ii) similar results more quickly. Numerous papers in the literature use this alternative to reduce computational costs [6–9].

This paper analyzes a structural optimization problem to evaluate the effectiveness of the multi-objective metaheuristic with iterative parameter distribution estimation (MM-IPDE) algorithm in conjunction with five different metamodels widely used in different applications (they will be presented in section 3). The optimization problem concerns minimizing the weight and maximizing the first critical load factor of a 120-bar truss. The axial stresses and nodal displacements are the constraints. First, a study of the training performance of the five metamodels is presented. Then, the results for truss optimization are presented, and an analysis is performed using two performance metrics. Finally, the best solutions are extracted and presented according to some importance set by the decision maker's (DM) preferences, using a multicriteria tournament decision (MTD) method.

The paper is organized as follows. Section 2 presents the formulation of the structural optimization problem and the algorithm used in the numerical experiments. The metamodeling approach used is described in Section 3. Section 4 presents the numerical experiments and an analysis of the results obtained. Finally, the paper ends with the conclusions in Section 5.

# 2 Multi-objective structural optimization

## 2.1 The structural optimization problem

A multi-objective structural optimization problem (MOSOP) has two or more objective functions to be minimized (or maximized) simultaneously. The MOSOP considered here is to find the set of cross-sectional areas $\mathbf{x} = \{A_1, A_2, ..., A_N\}$ that minimizes the weight of the structure and maximizes its first critical load factor as follows.

$$\min \quad W(\mathbf{x}) = \sum_{i=1}^{N} \rho A_i L_i, \quad \text{and} \quad \max \quad \lambda_1(\mathbf{x}), \tag{1}$$

subject to the normalized displacement constraints the normalized stress constraints

$$\frac{|u_j|}{\bar{u}} - 1 \leq 0, \quad j = 1, \ldots, M, \tag{2}$$

$$\frac{|\sigma_i|}{\bar{\sigma}} - 1 \leq 0, \quad i = 1, \ldots, N, \tag{3}$$

where $W$ is the weight of the structure, $\rho$ is the specific mass of the material, $L_i$ is the length of the $i-$th bar of the structure, $\lambda_1$ is the first critical load factor concerning the global stability of the structure, and $u_j$ and $\sigma_i$ are the nodal displacements of the $j-$th translational degree of freedom, and the stress of the $i-$th bar, respectively. $\bar{u}$ is the maximum displacement for each node, and $\bar{\sigma}$ is the allowable stress for the material. $M$ is the number of translational degrees of freedom, and $N$ is the total number of members in the truss structure.

## 2.2 The multi-objective DE algorithm

The differential evolution algorithm (DE), introduced by Storn and Price [10], is based on the generation and evolution of a population of candidate solutions with continuous variables. It is currently considered one of the most popular metaheuristics for solving optimization problems. The DE-based multi-objective structural optimization algorithms (MOEAs) used to solve the MOSOP formulated in this paper is the multi-objective metaheuristic with iterative parameter distribution estimation (MM-IPDE), proposed by Wansasueb *et al.* [11].

The MM-IPDE algorithm uses a concept of distribution estimation to tune metaheuristic control parameters and has become an efficient alternative for many types of optimization problems.

# 3 Metamodeling

Machine learning (ML) is a data analysis method that enables computers to mimic the natural learning processes of humans and animals by learning from experience. Instead of relying on a given equation as a model, ML algorithms use computational methods to ”learn” directly from data. In supervised machine learning, a model is created that makes predictions based on the available data and considers uncertainties. This supervised approach uses classification and regression techniques to develop effective ML models.

In this context, metamodeling is a technique often applied to optimization problems tackled with metaheuristics. Its primary goal is to identify and estimate the relationship between the inputs and outputs of a simulation model, which is used to evaluate potential solutions in the optimization process [12]. A metamodel effectively captures the relationship between the values of the decision variables and the simulation results and provides an approximation of the objective function with significantly reduced computation time compared to direct simulation.

A large number of ML techniques are used in a variety of applications, including artificial neural networks, gaussian processes, polynomial regression, ensembles (such as random forest and gradient boosted trees), and support vector machines [13].

In this study, five ML algorithms are applied to a regression problem involving the prediction of continuous variables: support vector machine (SVM), random forest (RF), gradient tree boosting (GTB), Bagging (BG), and XGBoost. Each of these metamodels has specific parameters that need to be selected since an appropriate choice influences the behavior and performance of these metamodels. Table 1 details the parameters used for each of the five algorithms.

The metamodeling steps proposed in this paper are, therefore as follows:

- **Step 1**: The Latin hypercube sampling (LHS) method [14] is used to generate four data sets (500, 1000, 1500, and 2000 samples) with the design variables (A) and then the finite element method (FEM) [15] is used to determine the first critical load factor ($\lambda$);

Table 1. Metamodels parameters.

| Metamodel | Parameter values |
|---|---|
| SVM | kernel='rbf', C= 10000, gamma= 0.1, epsilon= 0.001 |
| RF | n_estimators= 200, max_depth= 10, bootstrap=True, min_impurity_decrease=1e-07 |
| GTB | criterion= 'friedman_mse', n_estimators= 300, max_depth= 5, learning_rate= 0.01, loss='squared_error' |
| BG | estimator=svm.SVR(), n_estimators= 100 |
| XGBoost | booster='gbtree', objective='reg:squarederror', n_estimators= 500, max_depth= 7, learning_rate= 0.05, eta= 0.1, subsample= 0.7, colsample_bytree= 0.8 |

- **Step 2**: All data is split into a training set (85% of the data) and a test set (15% of the data);
- **Step 3**: The metamodel is trained using the parameters in Table 1;
- **Step 4**: The MM-IPDE algorithm is introduced to solve the optimization problem and the first critical load factor is predicted using the trained base model.

The steps outlined above are performed for each of the five metamodels considered here.

## 4 Numerical experiments

This section introduces the spatial truss to be optimized in this paper. The 120-bar truss is shown in Fig. 1 and is subjected to the MOSOP described in Section 2.1. For example, the truss's characteristics and descriptions can be found in [16]. In the experiments, the initial population was randomly generated, with the maximum number of objective function evaluations being 5000 (50 individuals and 100 generations), the number of independent runs being 20, and all presented solutions are rigorously feasible. The MM-IPDE algorithm was developed in MATLAB® and the Python library scikit-learn[1] was used for the metamodels, and the FEM was developed by the authors. The calculations were performed with a MacBook Pro - Apple M2 Pro with 16 GB RAM.

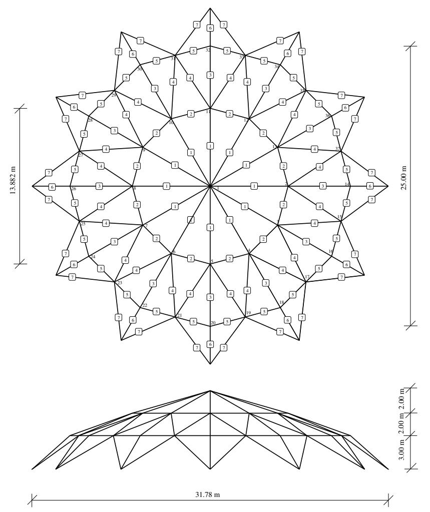


Figure 1. 120-bar truss, taken from Lemonge *et al.*[16].

### 4.1 Results and analysis

Four cases were selected from the training data using the LHS method, corresponding to the size of the training data set of 500, 1000, 1500, and 2000. In addition, four performance indicators of ML algorithms were calculated. The results are shown in Figure 2, such as mean square error (MSE) (Figure 2(a)), mean absolute error (MAE) (Figure 2(b)), mean absolute percentage error (MAPE) (Figure 2(c)), and coefficient of determination ($R^2$) (Figure 2(d)). In this figure, the horizontal axis refers to the samples.

The performance of all five algorithms was very good, as observed in the Figure 2. It can be seen that all metamodels produced small values for the performance indicators MSE, MAE, and MAPE, except for SVM,

[1] https://scikit-learn.org/stable/

which had slightly higher values. Although the sample size of 500 was small, the five ML algorithms achieved very high R2 values, reaching 84.92%, 99.51%, 99.04%, 93.56%, and 95.46% for SVM, RF, GTB, BG, and XGBoost, respectively. The high values of all these ML algorithms prove to be efficient and practicable.

In addition, the computational time in the training phase of the ML algorithms is shown in Figure 3. SVM required the least computational effort compared to the other ML algorithms. In particular, SMV required only 0.072, 0.044, 0.045, and 0.074 seconds, corresponding to training data sizes of 500, 1000, 1500, and 2000, respectively. The BG metamodel took 0.297, 0.994, 1.052, and 3.209 seconds, making it more expensive than all the other metamodels, although it performed well on the other metrics. Because of this, all five metamodels were considered in the structural optimization phase.

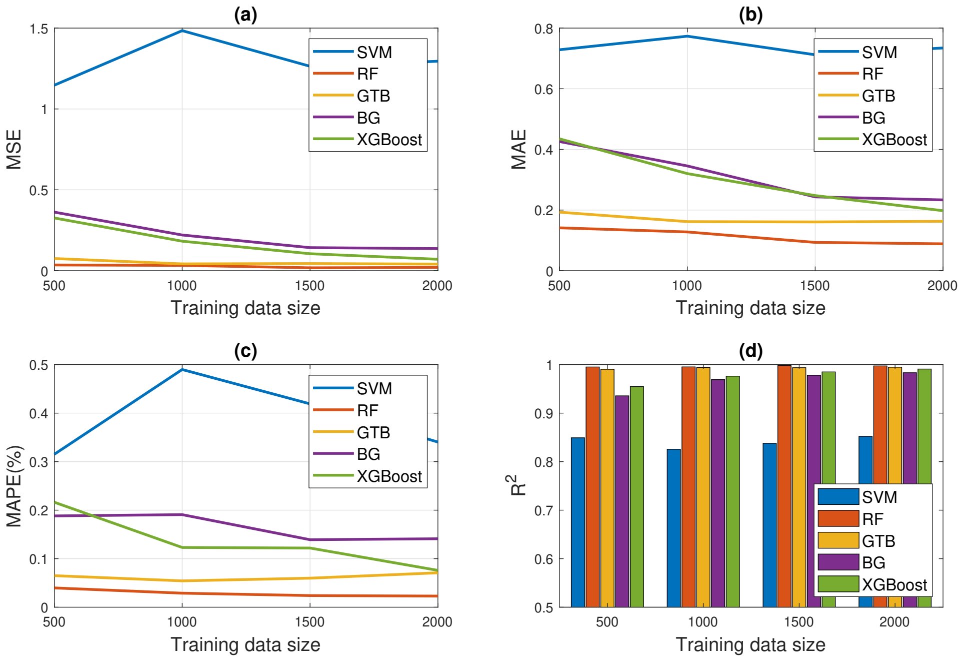


Figure 2. The mean squared error (MSE), the mean absolute error (MAE), the mean absolute percentage error (MAPE), and the coefficient of determination ($R^2$) of individuals metamodel in the training phase.

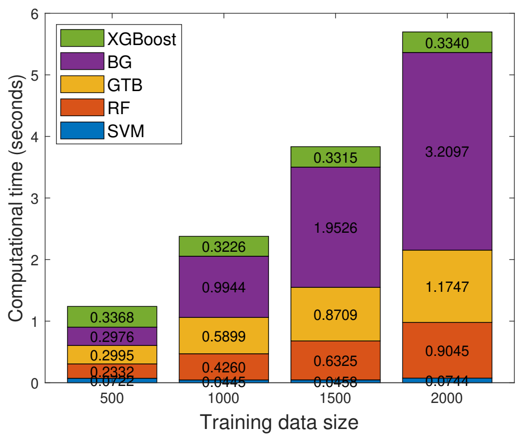


Figure 3. Computational time for the four data sets in the training phase.

In the next step, the optimization results (Pareto fronts) of the original MM-IPDE algorithm and with the coupled metamodels are shown in Figure 4, considering the data set of 500 (Figure 4(a)), 1000 (Figure 4(b)), 1500 (Figure 4(d)), and 2000 (Figure 4(d)) samples. In the figures and text, the acronym NM (no metamodel) will be used to refer to the MMIPDE algorithm without any coupled metamodel, and for each coupled metamodel to the MMIPDE algorithm, the reference will be made through the name of each metamodel. In this stage, the predictions of the metamodels determine the critical load factors, and the solutions of the multicriteria tournament decision (MTD) method are also shown in the figure. The MTD method was proposed by Parreiras & Vasconcelos [17] and is used to find the best solutions according to the importance (weights) set by the decision maker (DM). Two scenarios are used, considering two criteria: (i) the weight of the structure and (ii) the first critical load factor. The scenarios (Sc) are described: scenario 1 - all criteria have the same importance $(w_1, w_2) = (0.5, 0.5)$, and scenario 2 - criterion (i) is the most important $(w_1, w_2) = (0.75, 0.25)$. The square in cyan and the diamond in magenta represent these solutions.

The computational time of these runs is shown in Figure 5. The NM took 559.84s, and the BG took 522.97s, 583.90s, 634.34s, and 674.78s for the 500, 1000, 1500, and 2000 sample data sets, respectively. These are high values, and the BG is inefficient for the proposal. The other algorithms achieved this in a shorter time than the NM. For example, the SVM achieved 72.35s, 75.40s, 77.68s, and 77.90s for the 500, 1000, 1500, and 2000 sample data sets, respectively.

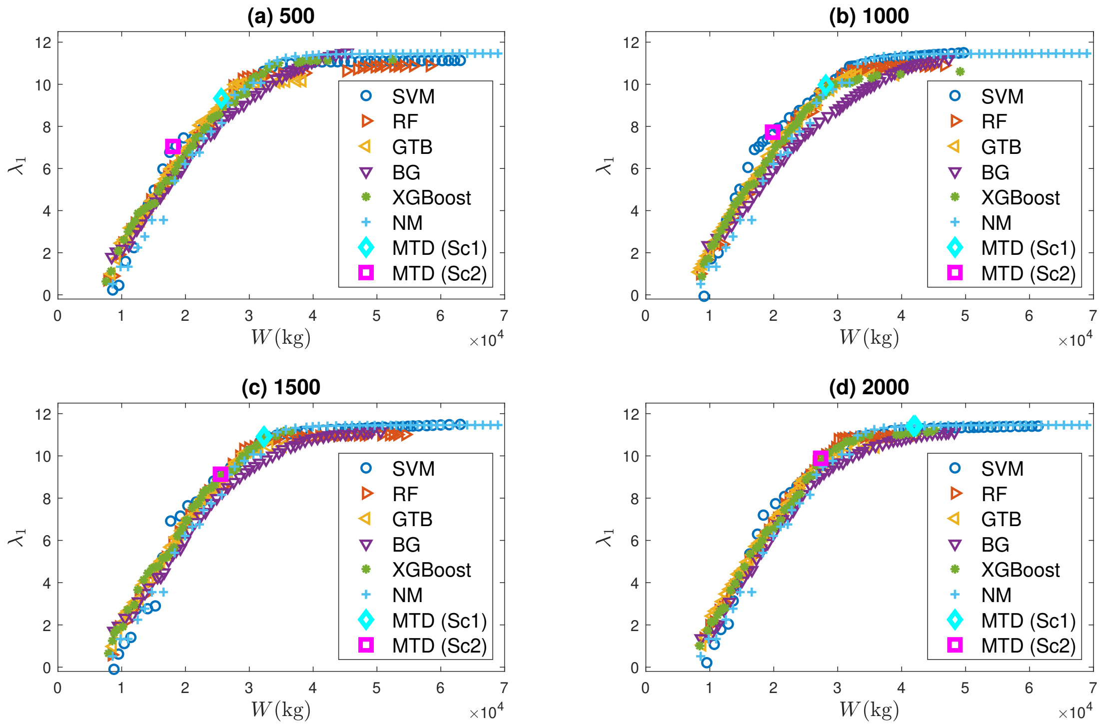


Figure 4. Pareto fronts of the NM, SVM, RF, GTB, BG, and XGBoost; and MTD solutions for the 120-bar truss.

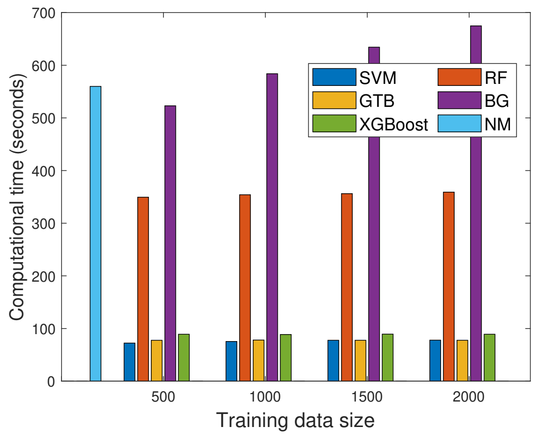


Figure 5. Total computational time of the 20 independent runs for each algorithm on each data set.

To measure the performance of the algorithms, the mean and standard deviation (std) of the hypervolume (HV) and inverted generational distance plus (IGD+) are shown in Table 2. Bold values represent the best results for each metric. For the mean HV, all algorithms performed well and achieved similar values for the four data sizes. The same applies to the std HV, where the algorithms achieved small and similar values. Particularly noteworthy are the SVM and RF metamodels, which achieved the best values for this metric. For the IGD+ metric, the NM achieved the three best values, followed by the RF metamodel. However, the performance of all algorithms was satisfactory for all analyzed data sets.

In a final analysis, the solutions obtained with the MTD method, shown in Figure 4, are presented in Table 3. NM was used in this comparison for the four data size in the two scenarios evaluated. The table shows the design variables (A) of each solution and their respective objective function values. The algorithm to which these solutions belong (Origin) is also given at the end of the table. That is, for MTD scenarios 1 and 2, the solutions come only from the RF, SVM, and NM.

Table 2. Mean and standard deviation (std) of the results obtained for HV and IGD+ from the 20 independent runs of the 120-bar truss.

| Data size | Metrics | NM | SVM | RF | GTB | BG | XGBoost |
|---|---|---|---|---|---|---|---|
| 500 | HV (mean) | 0.786505 | 0.782920 | 0.773055 | 0.747542 | 0.780324 | **0.791353** |
| | HV (std) | 0.004803 | 0.004478 | 0.003180 | **0.002103** | 0.005604 | 0.003605 |
| | IGD+ (mean) | 0.002341 | 0.008713 | 0.014383 | 0.035173 | **0.001747** | 0.005802 |
| | IGD+ (std) | **0.001135** | 0.001203 | 0.000629 | 0.000271 | 0.001580 | 0.002157 |
| 1000 | HV (mean) | 0.792170 | **0.818030** | 0.783674 | 0.765207 | 0.757950 | 0.776306 |
| | HV (std) | 0.004879 | 0.004048 | 0.002011 | **0.001778** | 0.005082 | 0.005105 |
| | IGD+ (mean) | 0.001135 | **0.000511** | 0.015705 | 0.031909 | 0.006119 | 0.017105 |
| | IGD+ (std) | 0.000644 | 0.000346 | **0.000151** | 0.000227 | 0.001716 | 0.005487 |
| 1500 | HV (mean) | 0.794940 | **0.808317** | 0.787943 | 0.768157 | 0.767858 | 0.801262 |
| | HV (std) | 0.004861 | 0.004256 | **0.002345** | 0.002752 | 0.003535 | 0.008391 |
| | IGD+ (mean) | **0.004609** | 0.004915 | 0.018733 | 0.041639 | 0.025036 | 0.011281 |
| | IGD+ (std) | 0.001351 | 0.000719 | **0.000317** | 0.000371 | 0.001140 | 0.006958 |
| 2000 | HV (mean) | 0.798332 | **0.812562** | 0.795624 | 0.770421 | 0.777547 | 0.806076 |
| | HV (std) | 0.004887 | 0.006494 | **0.001582** | 0.002025 | 0.003098 | 0.004347 |
| | IGD+ (mean) | **0.008505** | 0.008748 | 0.028095 | 0.056467 | 0.030764 | 0.017669 |
| | IGD+ (std) | 0.001570 | 0.001958 | 0.000339 | **0.000335** | 0.001054 | 0.004004 |

Table 3. Design variables (dv) and objective function values of the MTD solutions (Scenarios (Sc) 1 and 2) of the 120-bar truss. $W(\mathbf{x})$ in kg.

| Data size | 500 | | 1000 | | 1500 | | 2000 | |
|---|---|---|---|---|---|---|---|---|
| dv | Sc1 | Sc2 | Sc1 | Sc2 | Sc1 | Sc2 | Sc1 | Sc2 |
| $A_1$ | 0.013962 | 0.014000 | 0.013110 | 0.014000 | 0.014000 | 0.014000 | 0.014000 | 0.013458 |
| $A_2$ | 0.005010 | 0.005302 | 0.004642 | 0.003458 | 0.005218 | 0.005043 | 0.002195 | 0.004317 |
| $A_3$ | 0.001526 | 0.001581 | 0.001501 | 0.001782 | 0.001304 | 0.002011 | 0.007145 | 0.001561 |
| $A_4$ | 0.008819 | 0.002411 | 0.011352 | 0.003862 | 0.013994 | 0.008535 | 0.014000 | 0.010381 |
| $A_5$ | 0.002862 | 0.003089 | 0.003786 | 0.004103 | 0.004611 | 0.003374 | 0.014000 | 0.004258 |
| $A_6$ | 0.000878 | 0.001066 | 0.000384 | 0.001008 | 0.000314 | 0.000700 | 0.004362 | 0.000645 |
| $A_7$ | 0.000704 | 0.000487 | 0.000551 | 0.000450 | 0.000426 | 0.000404 | 0.000200 | 0.000380 |
| $W(\mathbf{x})$ | 25602.1088 | 18016.7392 | 28125.569 | 19810.466 | 32341.369 | 25475.732 | 41836.667 | 27329.308 |
| $\lambda_1(\mathbf{x})$ | 9.330 | 7.045 | 9.972 | 7.727 | 10.910 | 9.131 | 11.414 | 9.885 |
| Origin | RF | SVM | RF | SVM | SVM | SVM | NM | RF |

## 5 Conclusions

In this paper, we propose to apply five traditional machine learning algorithms, SVM, RF, GTB, BG, and XGBoost, as meta-models to nature-inspired metaheuristic algorithms for a regression problem to predict critical load factor values in a multi-objective structural optimization problem. The approach was shown to achieve good results, both in terms of objective function values (weight and critical load factor) and in terms of computational time reduction. The solutions obtained with the metamodels are competitive even without the use of the finite element method during the optimization process. The results were confirmed by using performance metrics both during the training of the metamodel dataset and during the evolutionary process.

For future work, we intend to apply the approach to solve other problems involving large optimization problems (larger number of bars, for instance, 120- and 582-bar truss) and expensive simulation models (nonlinear analysis). We also intend to investigate the effects of the parameters of the machine learning algorithms.

**Acknowledgements.** This study was financed in part by the Brazilian Agencies CAPES – Finance Code 001, CNPq (Grants 308105/2021-4 and 303221/2022-4), and FAPEMIG (Grant TEC PPM-00174-18 and APQ-00869-22).

**Authorship statement.** The authors hereby confirm that they are the sole liable persons responsible for the authorship of this work, and that all material that has been herein included as part of the present paper is either the property (and authorship) of the authors, or has the permission of the owners to be included here.

## References

[1] H. Assimi, A. Jamali, and N. Nariman-Zadeh. Multi-objective sizing and topology optimization of truss structures using genetic programming based on a new adaptive mutant operator. *Neural Computing and Applications*, vol. 31, pp. 5729–5749, 2019.

[2] J. P. G. Carvalho, É. C. Carvalho, D. E. Vargas, P. H. Hallak, B. S. Lima, and A. C. Lemonge. Multi-objective optimum design of truss structures using differential evolution algorithms. *Computers & Structures*, vol. 252, pp. 106544, 2021.

[3] J.-S. Chou and D.-N. Truong. Multiobjective forensic-based investigation algorithm for solving structural design problems. *Automation in Construction*, vol. 134, pp. 104084, 2022.

[4] É. C. Carvalho, J. P. G. Carvalho, H. S. Bernardino, A. C. Lemonge, P. H. Hallak, and D. E. Vargas. Solving multi-objective truss structural optimization problems considering natural frequencies of vibration and automatic member grouping. *Evolutionary Intelligence*, vol. , pp. 1–26, 2022.

[5] C. Blum and A. Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM computing surveys (CSUR)*, vol. 35, n. 3, pp. 268–308, 2003.

[6] J. V. S. Amaral, R. Carvalho Miranda, J. A. B. Montevechi, C. H. Santos, and G. T. Gabriel. Metamodeling-based simulation optimization in manufacturing problems: a comparative study. *The International Journal of Advanced Manufacturing Technology*, vol. 120, n. 7, pp. 5205–5224, 2022a.

[7] M. Sbayti, A. Ghiotti, R. Bahloul, H. BelhadjSalah, and S. Bruschi. Effective strategies of metamodeling and optimization of hot incremental sheet forming process of Ti6Al4Vartificial hip joint component. *Journal of Computational Science*, vol. 60, pp. 101595, 2022.

[8] I. Negrin, M. Kripka, and V. Yepes. Metamodel-assisted design optimization in the field of structural engineering: A literature review. In *Structures*, volume 52, pp. 609–631. Elsevier, 2023.

[9] V.-H. Truong, T.-S. Cao, and S. Tangaramvong. A robust machine learning-based framework for handling time-consuming constraints for bi-objective optimization of nonlinear steel structures. In *Structures*, volume 62, pp. 106226. Elsevier, 2024.

[10] R. Storn and K. Price. Differential evolution a simple and efficient adaptive scheme for global optimization over continuous spaces. Tech. Rep. 95-012, Univ. of California, Berkeley, CA, 1995.

[11] K. Wansasueb, N. Pholdee, N. Panagant, and S. Bureerat. Multiobjective meta-heuristic with iterative parameter distribution estimation for aeroelastic design of an aircraft wing. *Engineering with Computers*, vol. 38, pp. 695–713, 2022.

[12] A. Parnianifard, A. Azfanizam, M. Ariffin, and M. I. S. Ismail. Comparative study of metamodeling and sampling design for expensive and semi-expensive simulation models under uncertainty. *Simulation*, vol. 96, n. 1, pp. 89–110, 2020.

[13] J. V. S. Amaral, J. A. B. Montevechi, R. Carvalho Miranda, and W. T. Sousa Junior. Metamodel-based simulation optimization: A systematic literature review. *Simulation Modelling Practice and Theory*, vol. 114, pp. 102403, 2022b.

[14] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, vol. 42, n. 1, pp. 55–61, 2000.

[15] T. J. Hughes. *The finite element method: linear static and dynamic finite element analysis*. DoverPublications.com, 2012.

[16] A. C. Lemonge and H. J. Barbosa. An adaptive penalty scheme for genetic algorithms in structural optimization. *International Journal for Numerical methods in engineering*, vol. 59, n. 5, pp. 703–736, 2004.

[17] R. Parreiras and J. Vasconcelos. Decision making in multiobjective optimization aided by the multicriteria tournament decision method. *Nonlinear Analysis: Theory, Methods & Applications*, vol. 71, n. 12, pp. e191–e198, 2009.