applied sciences

MDPI

Article

Method for Parameter Tuning of Hybrid Optimization Algorithms for Problems with High Computational Costs of Objective Function Evaluations

Przemysław Sebastjan and Wacław Kuś *

Department of Computational Mechanics and Engineering, Silesian University of Technology, 44-100 Gliwice, Poland; przemek.sebastjan@gmail.com
* Correspondence: waclaw.kus@polsl.pl

Abstract: In this paper, the authors focus on presenting the methodology for tuning optimization algorithm parameters, with a special focus on evolutionary algorithm applications. The problem considered concerns the phenomenon of nonlinear buckling of the automotive shock absorber, which itself is solved using a commercial application of the finite element method (FEM) simulation. These analyses are usually time-consuming; therefore, the authors decided to use a surrogate model, which mimics the behavior of the actual nonlinear FEM simulation. Surrogate modeling (metamodeling) is utilized to drastically shorten the simulation time, and thus study numerous algorithm parameter combinations, allowing for tuning them and providing a robust and efficient tool for optimization. The example shown in this paper is related to the minimization of the shock absorber weight, taking into account the stability of the system. The presented method can be used in any optimization problem where the high computational cost of objective function evaluations prevents tuning of the algorithm parameters.

Keywords: optimization; metamodel; FEM; genetic algorithm; stability

check for updates

Citation: Sebastjan, P.; Kuś, W. Method for Parameter Tuning of Hybrid Optimization Algorithms for Problems with High Computational Costs of Objective Function Evaluations. *Appl. Sci.* **2023**, *13*, 6307. https://doi.org/10.3390/app13106307

Academic Editor: Gang Lei

Received: 18 April 2023
Revised: 11 May 2023
Accepted: 19 May 2023
Published: 22 May 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The optimization of shape and topology is one of the areas of rapid growth in the automotive industry. Many practical applications of sophisticated optimization algorithms have been published in the last years, such as the SIMP method [1], which uses relative densities of the material as design variables (therefore, the name solid isotropic material with penalization). Method dealing with similar limit load optimization problems utilizing the SIMP method has been described in [2], highlighting the drawbacks of using this scheme for buckling prediction problems. Another method called bi-directional evolutionary structural optimization (BESO) has been used in these applications, in accordance with [3]. One of the most recently developed methods for structural optimization is the level-set method (LSM), broadly described in [4]. The LSM does not use element-based densities as design variables; however, it changes the shape (the boundary) of the body based on the auxiliary field. Most of the popularized methods do not require explicit parametrization; therefore, their solution does not require the user to specify many parameters of the algorithm. On the other hand, parametric optimization, in which the parameters (variables) represent physical quantities, such as dimensions, material properties, loads, etc., requires a detailed definition of the optimization algorithm parameters. This may be seen especially in genetic algorithms, where population size (number of islands, number of individuals per island, number of generations) may be adjusted freely by the designer, in addition to the probabilities of migration, mutation, crossover, elite size, etc. Different combinations of the aforementioned algorithm parameters may show a broad scatter in the convergence rate or the final fitness function (objective function) during optimization. Evolutionary optimization is still an actively developing field. In [5], the authors proposed a modified version of the genetic

algorithm, in which the genetic operators are carried out independently on each generation, i.e., crossover is performed during every 2n generations, while the mutation during every 2n + 1 generations. The authors utilized a large number of individuals in the population combined with only a few generations. The great extinction genetic algorithm (GEGA) modification proposed in [6] incorporated nature-based phenomena, such as global disasters to remove part of the individuals from the population, which are later randomly replaced by new individuals. The authors focused on minimizing the number of objective function calls, which may be treated as a method to mitigate one of the major drawbacks of GA—excessive computational effort associated with simulation-based objective function evaluation. To capture the actual global optimum with limited resources, the gradient-based optimization procedure was incorporated after the evolutionary optimization. With a similar amount of individuals, yet utilized in different populations, the authors of [7] were able to optimize the automotive suspension layout with a limited number of generations and small population sizes in the Multi-Island version of the genetic algorithm (MIGA). A recent review of genetic algorithms [8] points out that the foreseen progress of GAs should cover the tuning of appropriate degrees of crossover and mutation operators. In addition, the hybridization of algorithms is marked as one of the potential solutions to the overly-intensified objective function calculations associated with the GAs. A combination of global-search GA with local-search techniques proves to provide an efficient optimization tool, assuring the balance between diversification and computational effort. The authors of [8] provide numerous references supporting the efficiency of this hybrid approach in numerous fields of science. In general, different algorithms can be joined sequentially or parallelly, making a hybrid optimization scheme. It is up to the designer which algorithm to apply and in which form (algorithm settings). There is generally no limit to algorithm combinations. In [9], the authors combined genetic algorithms with simulated annealing, while both of these algorithms are considered to be global-search strategies. The minimization of the compensation error of the magnetometer was addressed in their study, and the combination of population-based GA with single-individual SA helped in the achievement of this goal. The review of hybrid evolutionary optimization [10] shows that the hybrid approach suffers from similar problems as pure GAs. Finding the efficient population size, along with the proper tuning of genetic operators, is key. The authors pointed out that the mutation parameter seems to change its role in the hybrid approach (compared to solely GA approach), as the purpose of the GA is shifted toward exploration; therefore, the role of mutation should be to steer the algorithm from one basis to another, by adjusting its higher probability. Other hybridization options were highlighted, i.e., pairing the GA with an additional algorithm assuring the feasibility of solutions. In the case of LSM-based topology optimization, this may be achieved by the additional, one-variable optimization of the threshold parameter. However, this method could increase the final optimization time when the univariable optimization is carried out on a very nonlinear and multimodal landscape. In real-world industrial conditions, the time or computational effort available for optimization is limited. In these cases, finding the actual optimum may not be possible; therefore, the designers' task can be reformulated into finding the best quasi-optimum in the given time or using the available computational resources. Then, having an algorithm with well-tuned parameters is beneficial and desired, as presented, e.g., in [11,12]. How to obtain the proper parameter values is one of the engineering challenges, especially when the solution requires time-consuming FEM-based simulation. In these cases, it is beneficial to use the surrogate modeling approach to replace costly numerical simulations with quick metamodel calculations. This approach is the aim of the authors' research and will be explained in detail in further chapters.

Surrogate modeling, also referred to as metamodeling, is a constantly growing field of science. It allows not only for the replacement of the underlying, costly simulations and physical experiments, but it may also give some insight into the properties of the simulated or tested process, especially when dealing with normalized values, as the coefficient of the metamodels (as the polynomial regression model) represent the sensitivity or importance

of particular terms or parameters [13]. Recent review papers [14,15] show that the majority of the literature studies involve three types of metamodels: The artificial neural networks (ANNs), the Kriging, and the polynomial regression, which paired with statistical analysis is referred to as the response surface method (RSM). These surrogates account for 24, 15, and 17% of all surrogates in the recent papers. Moreover, the combination of metamodeling, along with hybrid optimization, is observed as an increasingly common practice among researchers. All of the aforementioned models were proposed in the XXth century; however, the recent development is focused, e.g., on reducing the number of hyper-parameters needed to be optimized during the Kriging fitting process or architecture of the ANNs. More complex Kriging models, such as the Universal Kriging incorporating the trend model, even though more sophisticated, do not always bring significant improvement due to limitations associated with the response of the underlying regression model (the trend). Nevertheless, part of the research is devoted not necessarily to the metamodels themselves, but to techniques that minimize the number of costly training samples needed to fit an efficient surrogate, or even to guide the sampling procedure based on areas of metamodel high uncertainty [16]—this technique is known as the infill process. Nonetheless, numerous papers utilize surrogates to replace costly computer simulations, e.g., as the authors of [17] minimized the mass of the jet engine components using ANN, which was carried out 40% faster than the optimization carried out on the numerical-analytical model. Moreover, the authors incorporated the infill process during the optimization. In [18], the Kriging method was used to solve the vibroacoustic problem when the objective function evaluation using the simulation was quite costly to perform the standard optimization. In [19], the authors incorporated the RSM to optimize the titanium alloy sheet forming process for medical applications, similar to the study in [20], where the energy system optimization was conducted using the RSM.

The paper discusses and presents methods typical for problems where the calculation of the objective function is very time-consuming. Typically, the biggest problem is the determination of the parameters of the optimization algorithms. This is in most cases carried out by running the optimization algorithm multiple times with different parameters. Unfortunately, it is inefficient and the total calculation time is sometimes unacceptable. Using the methodology presented in the article, it is possible to select the parameters of optimization algorithms at a significantly lower cost, which allows for an effective, time- and price-acceptable approach to solving the optimization problem, utilizing the metamodeling technique. Metamodeling itself is usually used to replace the computer simulation during the optimization process, but for extremely nonlinear and multimodal problems, a sufficiently precise metamodel may not exist. That is why the authors use the surrogates to mimic the behavior of these complex systems and to perform the tuning of the optimization algorithm parameters, which will then be used to perform constrained mass minimization on FEM models.

The description of the optimization problem is given in Section 2, highlighting the design variables, optimization objectives, and limitations of the FEM-based process. Section 2 considers the methods used to resolve the optimization problem, including the metamodeling part. The sampling plan for the surrogate training and validation process, metamodel types, and quality metrics are also described. The last part of Section 2 concerns optimization algorithms and their parameters, which will be subjected to the tuning process. The first part of Section 3 shows the results of the metamodeling approach, including a comparative study of the quality metrics. Then, the efficiency of optimization algorithms is shown depending on the algorithm parameters, which are then tuned to provide the most efficient optimization scenario. Section 3 is concluded with an example of the FEM-based optimization utilizing the best and worst algorithm parameter combinations for comparison and benchmark purposes. All conclusions and results are summarized in Section 4.

2. Materials and Methods

The shock absorber is a part mounted in every vehicle's suspension system. Its role is not only to provide the damping response, but also frequently to prevent the wheel from colliding with the (car) body. Therefore, it must withstand severe axial loads, especially when the vehicle goes through an obstacle, such as a pothole. As the energy generated by this impact is usually very high, the loads transferred by the shock absorber may cause it to bend, crack, or buckle, the last being addressed in this paper. In addition, the loads coming from the assembly process or the modular spring (Figure 1a) may induce significant stress in the structure, affecting the overall strength performance. The engineer's work is to reduce the mass of this component, utilizing one of many topology or shape optimization techniques.

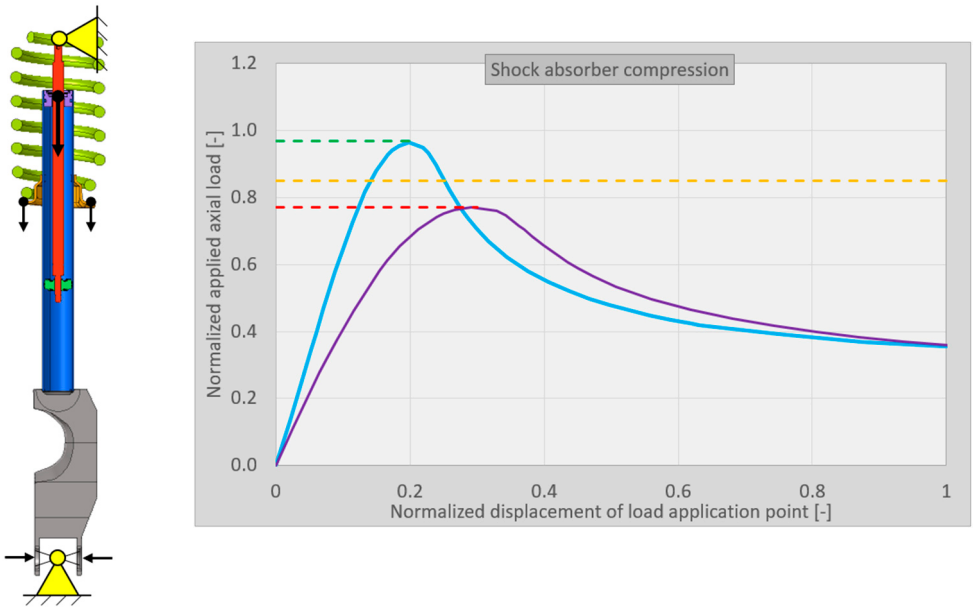


Figure 1. Strength calculation: (**a**) Boundary conditions, (**b**) considered stability problem (presented as load vs. deflection curves).

During the design phase, it is important to predict the proper mechanical response and to ensure that the damper will survive the given impact. It can be achieved by constraining a minimum force at which stability is lost, as shown in Figure 1b. As a result, the following optimization problem may be formulated:

Objective function f:

$$f = \min_{\mathbf{KV}} \rho \int_V \mathrm{d}V, \tag{1}$$

subjected to constraint q:

$$q(\mathbf{KV}) > 0,\ q(\mathbf{KV}) = F_{REQUIRED} - F_{LIMIT}(\mathbf{KV}), \tag{2}$$

where

$\mathbf{KV} = [KV_1, KV_2, \ldots, KV_N]$ is a vector of design variables controlling the shape of the structure;

ρ—material density;

V—the volume of the part being optimized;

F_{LIMIT}—maximum force achieved by the shock absorber before losing stability;

$F_{REQUIRED}$—the minimum force at which the shock absorber is allowed to lose stability.

The method for obtaining the value of the limit load F_{LIMIT} is usually a simulation using the finite element method. In each iteration of the design process, this constraint must

be fulfilled. As the mass of the part is minimized, the fulfillment of the stability constraint is ensured by the optimization algorithm. On the other hand, the mass optimization (minimization) task may be resolved in many ways; however, the authors focus on the application of the derivative-free level-set method applied for nonlinear 3D structures. The shape of the part is controlled by changing the design variable vector **KV**, which affects the distribution of the auxiliary scalar field Φ. For each of the finite elements, the value of the calculated field Φ is compared with the assumed threshold value in the following way:

$$\begin{cases} \Phi \geq threshold \rightarrow material \\ \Phi < threshold \rightarrow void \end{cases}, \tag{3}$$

Using this attitude, the material can be removed from the structure by adjusting the knot values at each of the radial basis function centers. A more detailed explanation of this method can be found in [21], since the authors developed a methodology suitable for the optimization of unstable structures. The exemplary distribution of the knots is shown in Figure 2.

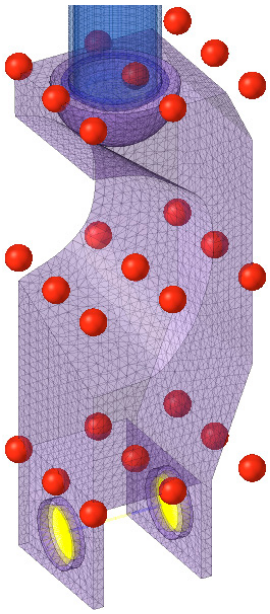


Figure 2. The distribution of the radial basis function centers—knots.

The main disadvantage of this approach is the time-consuming nonlinear finite element analysis needed to obtain information about the stability response of the system, for each design vector. Therefore, the authors decided to use surrogate models to represent the actual nonlinear system. This allowed for the verification of hundreds of thousands of design combinations in a small time fraction compared to a single FEM simulation, for the purpose of obtaining the best parameters of the optimization algorithm (tuning optimization algorithm parameters).

The finite element method was used to generate the samples that were then used to feed the metamodels (surrogate models). Two data sets were created to train and validate the metamodels—artificial neural network and Kriging.

As stated above, the method for obtaining the structural response of the system was the simulation using the finite element method (FEM). Due to the fact that the problem incorporates large displacements (geometrical nonlinearities), the interaction between bodies (contact), and the elastic–plastic material constitutive models (physical nonlinearities), the solution was derived using the commercial software, Abaqus/Standard [22]. The method for obtaining the new shape for FEM analysis was developed by the authors and consisted of the Python routine, which modified the Abaqus input deck based on all of the 27 optimization input parameters. These parameters are the coefficients of the radial basis

functions with fixed center positions. Based on the calculated auxiliary field values and assumed threshold, each of the finite elements was judged as active or disabled (taking into account the removal of prior numerical analysis) and a new numerical model was created. In each FEM analysis, the pinned boundary conditions were used at both ends of the shock absorber. In the first step of the simulations, the assembly loads were applied (bracket clamping and suspension spring preload). The incremental compression bumper load was applied in the second step, up to the point at which the structure is losing stability.

The considered example of the optimization consisted of 27 design parameters (variables), as shown in Figure 2 with RBF knots. The sampling plan in this case must be carefully chosen, in order to not force the design matrix to be quite excessive and to cover the design space in the most efficient manner. The main limitation is the total number of function calls (simulations) that can be used to create the sampling plan. Based on the available resources and the average calculation time, the maximum number of samples could not exceed 6750. Additionally, the sampling plan was divided into a 4:1 ratio; therefore, for each four training samples, one validation sample was kept separately for further surrogate quality assessment (validation). In this case, two plans were created:

- 5400 samples for the training set and 1350 samples for the validation set.
- 1350 samples for the training set and 340 samples for the validation set.

In both cases, the design matrices were independent of each other, i.e., two sampling plans were created separately for training and validation data. With the first sets ready (5400 training and 1350 validation samples), an additional small set was created to investigate the effect of sample size on the quality of the metamodel. Therefore, a new sample plan with 340 samples was used for validation purposes, whereas the previous validation set of 1350 samples was used as a training set. In total, three sampling plans were used to create two sets of data for metamodeling purposes.

There are numerous sampling plans available in the literature, among which full-factorial design of experiment (DOE), fractional–factorial DOE, Latin hypercube sampling (LHS), optimal Latin hypercube sampling (OLHS), orthogonal arrays, Sobol sequences, Box–Benhken sampling, or Monte Carlo methods are commonly incorporated in literature studies. The DOE approaches force the design matrix to grow exponentially, namely, full-factorial DOE for 27 factors on two levels requires 2^{27}, i.e., over 134 million analysis runs. In addition, only extreme values of the parameters are investigated; therefore, the actual nonlinear relations for parameters within their variation range can be lost. The schematic representation of different sampling plans for a small set of 2D data is shown in Figure 3. In the case of the factorial design, a uniform spread of design points is observed, even though the projection of the samples onto the axes is sparse. On the other hand, the Monte Carlo approach yields quite uniform distribution of samples projected on the axes, yet there may be regions of high density of design points, and regions without any points at all. The best saturation of design points projected onto the axes is visible for the LHS designs. Moreover, the OLHS maximizes the uniformness of design points distribution over the design area, in order that no overlapping points nor unsampled areas are observed when utilizing this sampling technique.

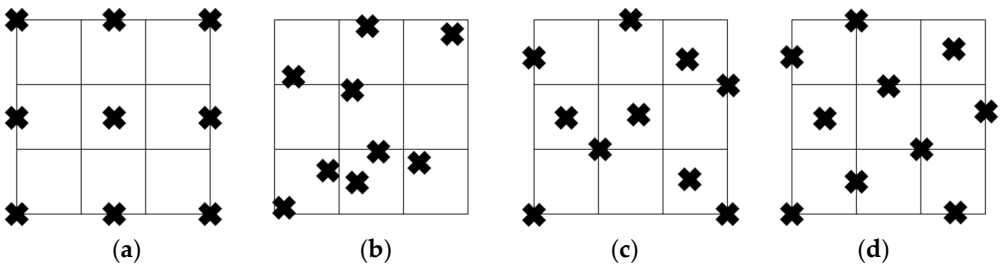


Figure 3. Schematic representation of different sampling plans: (**a**) Three-level factorial design, (**b**) Monte Carlo, (**c**) LHS, (**d**) OLHS.

To overcome the excessiveness of a sampling plan, and to achieve multidimensional stratification (i.e., a situation in which projections of individual samples on the axes are spread uniformly), the Latin hypercube sampling is introduced. Generally, the LHS is generated in a random manner, and thus the uniform space-filling property may not be reached. Therefore, the optimal Latin Hypercube Sampling was chosen as a source of the sample data. The difference between LHS and OLHS is the optimization phase, in which the even spread of points is achieved using the $\varnothing_p$ criterion, which is based on the maxmin distance criterion (minimum distance between any two sample points is maximized), as described in [23]:

$$\min_{l \le 1, j < n, i \ne i} d(x_i, x_j), \tag{4}$$

where $d(x_i, x_j)$ is the distance measured for any two sample points x_i and x_j, calculated as:

$$d(x_i, x_j) = d_{ij} = \left[\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^t \right]^{\frac{1}{t}}, \tag{5}$$

with $t = 1$ or $t = 2$ depending on the norm used (1 for the Manhattan norm, and 2 for the Euclidean norm).

Once all distances are calculated, they are sorted in ascending order $(d_1, d_2, \ldots, d_s)$ forming a distance vector. At the same time, the index list is created $(J_1, J_2, \ldots, J_s)$, which consists of J_j elements specifying how many pairs of the sampling plan are separated by distance d_j. Then, the optimal sampling plan is achieved if it minimizes the following criterion [24]:

$$\varnothing_p = \left[\sum_{i=1}^{s} J_i d_i^{-p} \right]^{\frac{1}{p}}, \tag{6}$$

OLHS assists in generating more evenly spread plans, which is not only better for the generality of the sampling plan, but also resolves problems with fitting the Kriging model, which will be discussed in more detail in the further part of the paper. For the training and validation data sets, separate OLHSs were created, each satisfying its own $\varnothing_p$. The matrix optimization was terminated once the assumed time was reached.

Two types of metamodels were used in the study, based on their ability to approximate (or interpolate) complex continuous and nonlinear functions without any prior assumptions regarding the function shape (which might be the case for polynomial models) and due to their universality. Studies utilizing similar metamodels were carried out, e.g., in [25].

The first metamodel is an artificial neural network (ANN) in the multilayer form. The foundations for this model were created in the 1940s by Warren McCulloch and Walter Pits [26] as a single artificial neuron (Figure 4a). Gathering multiple neurons in several layers together formed a simple neural network (Figure 4b), in which popularity started to grow significantly after the backpropagation learning algorithm was implemented [27].

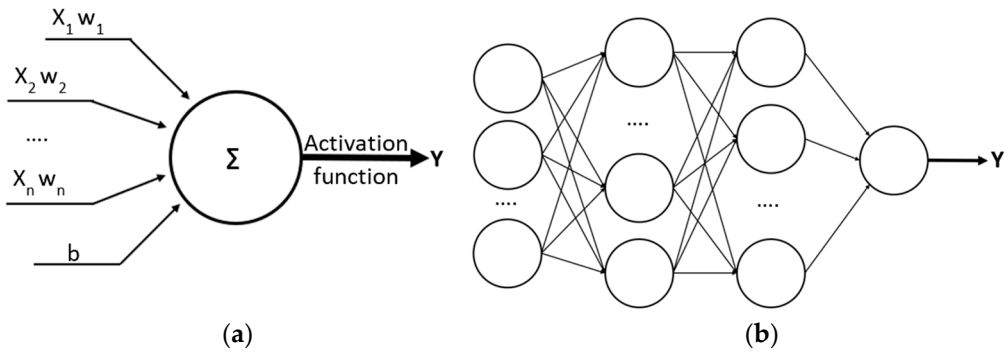


Figure 4. Neural network concepts: (**a**) Simple perceptron, (**b**) multilayer perceptron network.

There are many variants of ANNs, among which networks with radial basis function in the hidden layer are commonly used for metamodeling purposes [28,29], due to their clear architecture and simpler learning process. On the other hand, multilayer perceptron networks may have a complex architecture with numerous activation functions, hidden layer numbers, and features, such as regularization [30,31] or output normalization. In addition, there is a broad spectrum of weight and bias optimizers, among which stochastic gradient descent (SGD), SGD with momentum (e.g., NADAM—Nesterov adaptive momentum), and limited memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithms are one of the most common and well-known in the literature [32]. This complexity may seem as an obstacle; however, it provides a strong, flexible tool to adjust the network to achieve its highest efficiency. The main disadvantage, in this case, is the time-consuming process of finding the optimal network architecture and its set of properties. Once the artificial neural network is trained on a given set of data, the estimation of the objective function using unknown vectors is carried out using the optimized weights and biases from the training procedure. The implementation of ANN and FEA in problems similar to the one presented in this paper was described in [33], where the critical loads for the structures were investigated. The authors created the ANN metamodel using the python programming language (PyTorch) framework, which enables efficient parallelization, assisting in the reduction in model training time significantly.

The second metamodel is called the Kriging, after Daniel G. Krige, who developed this interpolation metamodel for geostatistics as his master thesis [34]. Although it was designed for mining purposes, such as gold exploration using sampling in a two-dimensional space (map), the method may be adapted for spaces with more dimensions. Kriging models are widely used as surrogate models for costly FEM-based simulations, even when severe nonlinearities are present [35,36].

There are several types of Kriging models, including simple, ordinary, universal, and blind [37], with the first being used in this paper for its generality and ease of implementation (sci-kit learn Python framework). The general form of the Kriging is the following:

$$Y(x) = f(x) + Z(x), \tag{7}$$

where:
$Y(x)$—predicted value;
$f(x)$—trend function;
$Z(x)$—realization of the normally distributed Gaussian random process.

Most commonly, the Gaussian correlation function is incorporated in the elements of the covariance matrix (defining the $Z(x)$):

$$\psi^{(i)} = e^{-\sum_{j=1}^{k} \theta_j \left| x_j^{(i)} - x_j \right|^2}, \tag{8}$$

where θ_j is theta parameter, scaling the width of the basis function, and k is the number of design variables (dimensions).

Once the training data are shown to the Kriging model, the hyperparameters of the basis functions are optimized by maximizing the ln-likelihood function. The predictions on the unseen data are calculated using both the training data set and the new, unseen data.

Based on the previous studies, the authors decided to focus on a combination of two algorithms, both being a part of the evolutionary optimization group, the genetic algorithm in its multi-island version (MIGA), and the evolution strategy (EVOL). This mixture of algorithms is not unique, as other tailor-made optimization schemes utilize a similar strategy [38].

Gradient methods were not included in this study, as they are not as efficient in optimizing systems, the response of which can be seen as quantified, i.e., the objective function is not continuous in the whole parameter domain, but it has constant values for wide ranges of parameter values, making the gradient calculation incorrect, as presented in Figure 5.

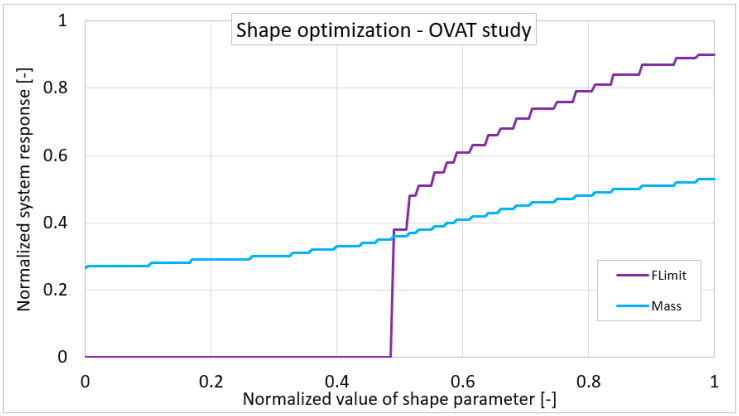


Figure 5. One variable at the time of study—influence of shape parameter (knot value) on mass and limit load of the system.

The multi-island version of the genetic algorithm is an extension of the work carried out by J. Holland in 1975. Compared to the basic version of the algorithm, the MIGA consists of several autonomous populations, referred to as islands [39]. Genetic operations, such as selection, cross-over, and mutation are performed (with a certain probability) only among individuals from a single island. Once a fixed number of generations is achieved (migration interval), migration occurs with a certain percentage amount of individuals from each island (migration rate). As an effect, the populations are mixed, providing more diverse and probably more fitted individuals. Elitism is introduced, assuring that the best individuals are not excluded from the population by genetic operators, and thus are preserved throughout the algorithm operation.

The evolution strategy (EVOL) is another variant of biology-inspired algorithms, in contrast to genetic algorithms, as the mutation is the only operator used to create new generations [40]. The rate of mutation is automatically controlled, based on the number of (un)successful mutations in past generations, making the process self-adaptive. In this paper, the EVOL algorithm is a 1 + λ variant, which means that λ offsprings are generated by mutation, and the best-fitted of these λ individuals becomes the parent for the next generation.

The aim of the algorithm parameter optimization is to maximize the optimization performance with a limited number of FEM simulations. The total number of function calls during the bracket mass optimization (described in the second chapter) is restricted to 2000, due to the time constraint considered in the paper. Therefore, all of the studies performed on surrogate models were also limited to 2000 calls, from which 80% (1600) were used for the genetic algorithm and 20% (400) for the evolution strategy, creating the hybrid optimization flow, as shown in Figure 6.

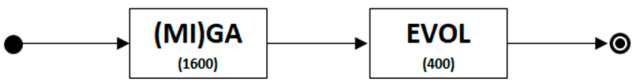


Figure 6. The hybrid optimization flow.

With the fixed optimization scheme, one can start to adjust the parameters of the optimization algorithms. To investigate the influence of these parameters (and their interactions) on the final objective function value, the authors decided to use a statistical approach, namely, design of experiment (DOE). The total number of analyzed parameters in the case considered in this paper is eight; however, the study is divided into two groups to reduce the size of the testing plan. In the first study, the single-population genetic algorithm was examined, along with evolution strategy parameters, and, based on the outcome of the analysis, the optimal values (one minimizing the objective function) were used in the second study of the multi-island version of the GA. More details are presented in the following section, highlighting the hybrid algorithm parameters and their levels.

The hybrid approach with a genetic algorithm that incorporates a single population (the most widely used version of the genetic algorithm—GA)—Table 1.

Table 1. Parameters for DOE performed on the GA and EVOL.

No.	Symbol	Parameter Name	Min	Max
1	A	Population ratio R	0.16	1.00
2	B	Crossover rate	0.50	1.00
3	C	Mutation rate	0.01	0.10
4	D	Penalty multiplier	10.0	1000.0
5	E	Penalty exponent	1.0	3.0

Parameter A—Population ratio R—is an artificial parameter to control the ratio between the number of individuals and the number of generations that make up the total population. It is simply calculated as follows:

$$\text{Population Ratio R} = \frac{\text{Number Of Individuals}}{\text{Number Of Generations}} \quad (9)$$

Knowing that the total number of function calls (individuals in the whole GA process) is fixed at 1600, one can easily modify the population architecture using the Population Ratio parameter.

The hybrid approach with a genetic algorithm having multiple populations (multi-island version of the algorithm—MIGA), with genetic operators and penalty function parameters fixed based on results from the first study and parameters given in Table 2.

Table 2. Parameters for DOE performed on the MIGA.

No.	Symbol	Parameter Name	Min	Max
1	A	Number of islands	2	4
2	B	Number of generations	40	100
3	C	Migration rate	0.05	0.20
4	D	Migration interval	4	20

Parameters D and E from the first table—Penalty multiplier and Penalty exponent—are the only parameters linked to both of the considered optimization algorithms, where the penalty function is added to the objective value once any of the constraints are violated. The level of penalization may or may not prevent infeasible designs from being used for further genetic operations, depending on the penalty characteristic. The penalty formula is the following:

$$\text{Penalty} = \text{Penalty}_{\text{base}} + \text{Penalty}_{\text{multiplier}} \times \text{Violation}^{\text{Penalty}_{\text{exponent}}} \quad (10)$$

There are variants of penalty function that include the constant term for infeasible designs, but they may act quite conservatively at close proximity to a feasible region; therefore, in this study, the constant term was excluded.

3. Numerical Results

The results are divided into several, subsequent sections, covering step-by-step each of the fields of interest. The effects of the metamodels training are described in Section 3.1. Tuning the optimization algorithm parameters utilizing the DOE methodology is shown in Section 3.2. Finally, the results of the new methodology implemented in the FEM-based optimization scenario are shown in Section 3.3.

3.1. Metamodeling Results

Metamodels described in the previous section are compared using the coefficient of determination R^2, using the Formula (11). The R^2 results for different neural network

topologies were plotted in Figure 7, showing the influence of the number of hidden neurons in each layer on the final metamodel performance. Two types of metamodels are then compared in Table 3.

Table 3. Resultant R^2 for validation data—best-fit models.

Response	F_{LIMIT}		Mass	
Model	**ANN**	**Kriging**	**ANN**	**Kriging**
Big set	0.765	0.716	0.942	0.960
Small set	0.698	0.620	0.926	0.940

$$R^2 = 1 - \frac{\sum_{i=0}^{n_t}\left(y^{(i)} - \hat{y}^{(i)}\right)^2}{\sum_{i=0}^{n_t}\left(y^{(i)} - \bar{y}\right)^2} = 1 - \frac{MSE}{Var(y)} \tag{11}$$

where:

n_t—the number of points for which the metamodel prediction is held out;
$y(i)$—the i-th response value obtained using FEM;
$\hat{y}(i)$—i-th response value predicted with the metamodel;
$\bar{y}$—mean value from all $y(i)$ (FEM).

As R^2 for training data is not assuring that the metamodel can predict unseen data correctly, only the R^2 values of validation data were used for comparison. Moreover, when the coefficient of determination is high for training data and, at the same time, low for validation data, it usually means that the model is overfitted, making it useless for any applications. The same studies were conducted for both sets of data: Big (5400 training, 1350 validation samples) and small (1350 training, 340 validation samples) to investigate the effect of sample size on the precision of surrogate model predictions. Furthermore, the metamodel training time is significantly reduced with the decreasing data size.

Artificial neural networks with different topologies, activation functions, and/or regularization methods were compared, and the best-fitted ones were reported herein—see Table 3 for details. Each combination was run 10 times to partially overcome the effect of randomness, associated with the generation of initial weights and biases. As a result of the model fitting, the ANN used for FLIMIT prediction, had 27-9-39-1 architecture, with the rectified linear unit (ReLU) activation function in both hidden layers. The influence of the network topology on the coefficient of determination (for validation data) R^2V is shown in Figure 7.

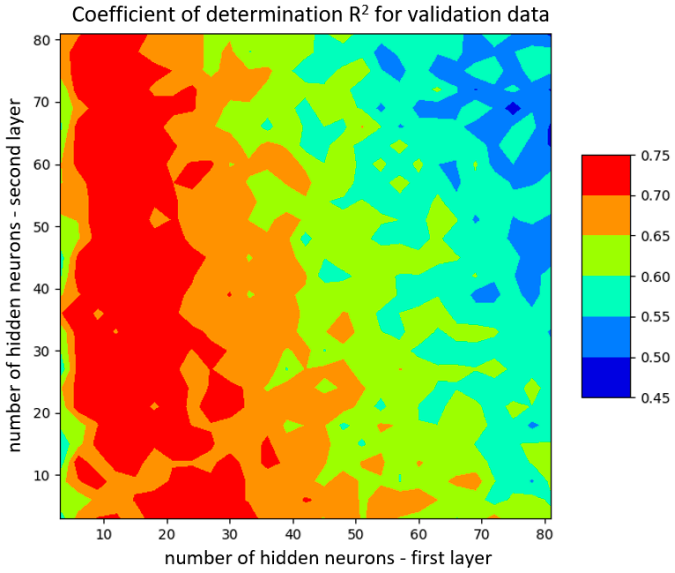


Figure 7. Coefficient of determination R^2V for validation data.

In general, the ANN models were better fitted once the bias and weight optimization (training) was performed incorporating the L-BFGS algorithm, rather than NADAM and SGD. Furthermore, L-BFGS converged significantly faster (40–100 iterations) compared to SGD (10,000–20,000 iterations). The only drawback is higher memory consumption associated with gradient history; however, the total usage is still orders of magnitude lower compared to the FEM calculations. The convergence history of the ANN training process is shown in Figure 8.

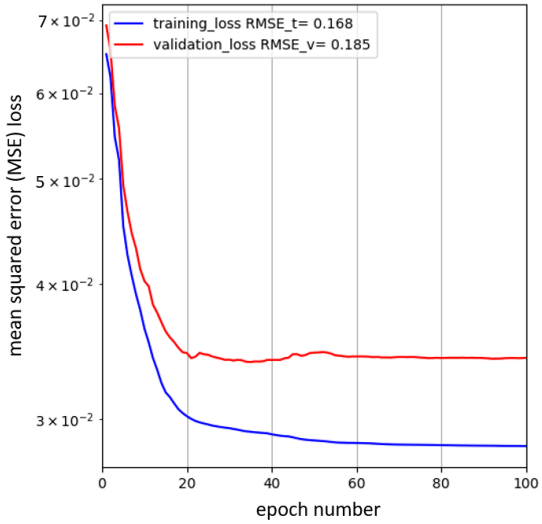


Figure 8. Training and validation loss as a function of epochs.

On the other hand, the simple Kriging method was taken as a model of choice to be used for mass predictions, as it outperformed the neural networks (in predictive accuracy) for this particular output. Moreover, similar to ANNs, each Kriging fitting was repeated 10×—once with equal initial theta values, and 9× with random initiation of theta vector.

The summary of the R^2 values is shown in Table 3. Values highlighted in bold font in the table represent the metamodels used for further studies on hybrid optimization algorithm parameters verification.

3.2. Design of Experiment—Hybrid Algorithm Parameters Optimization

The full factorial designs are widely used in similar studies [41]. On the other hand, fractional factorial DOEs greatly accelerate the analysis due to a limited amount of the required data (fraction of a full-factorial design), and the aliasing may be acceptable, knowing that high-order interactions between parameters are rarely observed [42]. However, the authors found that even though the mutual interactions are indeed limited, the nature of the main effects shows strongly nonlinear behavior. Therefore, rather than full- or fractional−factorial designs, the authors decided to utilize the full-quadratic response surface method (RSM) with Box−Behnken (BB) sampling. This type of sampling allows one to estimate the effect of each factor on three levels: −1, 0, and +1 (using coded units), without the need to run the three-level full-factorial design.

3.2.1. Single-Population Genetic Algorithm and Evolution Strategy Results

The factors that were found to have the highest impact on the resultant optimization capabilities are the mutation rate, penalty exponent, and penalty multiplier—C, D, and E from Table 1. Other factors are assumed to be less important, i.e., their effect on the objective function minimization is statistically insignificant. The main effect plot (for averaged responses) is shown in Figure 9, using the fitted full-quadratic response surface (RSM).

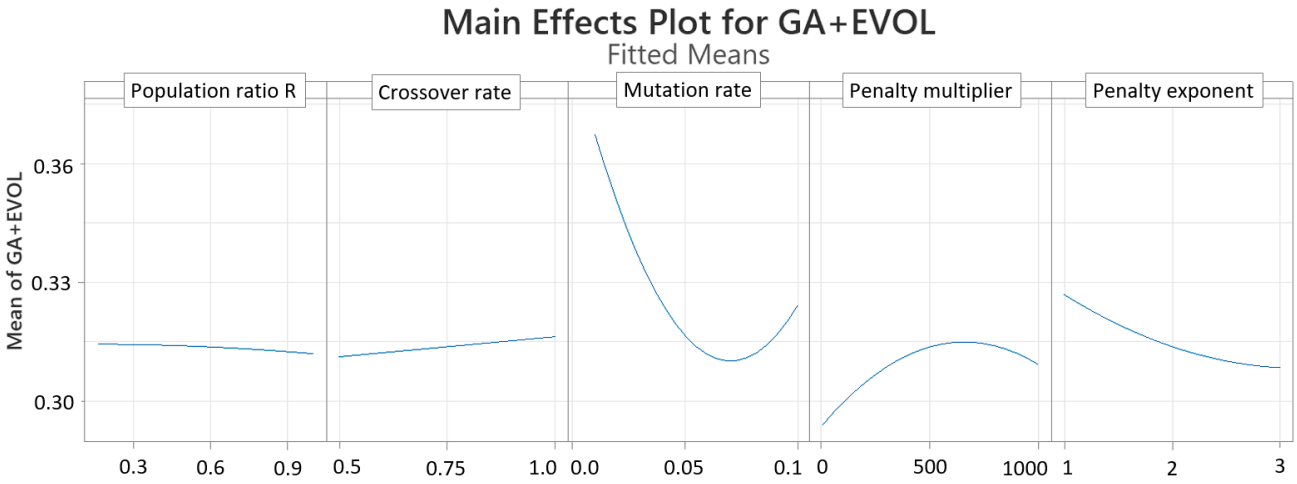


Figure 9. The main effect plot for the hybrid optimization using GA + EVOL approach (mean value of the fitted RSM shown in the Y axis).

The factor associated with the population layout, as well as the crossover rate, show a nearly linear and barely noticeable effect on the algorithm efficiency. On the other hand, a very significant impact is seen for the strongly nonlinear effect of the mutation rate. Both factors connected to the penalty function are nonlinear in their effects on the mean objective function, but their impact is not as strong as the mutation rate in a given example (even though they are still considered significant factors).

The penalty function, which is studied in the literature significantly less often, compared to other evolution algorithm parameters, shows high mutual interactions between the multiplier and the exponent, as shown below. The increasing multiplier (from 10 to 1000) may diminish the efficiency of the algorithm—when the exponent is equal to 3 or improves the efficiency when the exponent is equal to 1. This shows that penalization may be one of the critical aspects of constrained evolutionary optimization (Figure 10).

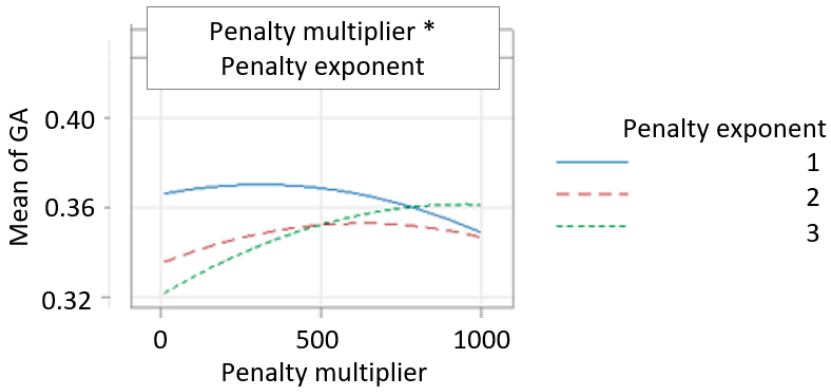


Figure 10. The interaction between the penalty multiplier and the penalty exponent and its effect on the objective function.

In order to utilize the maximum efficiency of the hybrid evolution algorithm, the optimization of the parameters of the factors was conducted in Minitab 21.1 [43]. The effect of the parameters adjustment is presented in Figure 11. A small-sized population of 16 individuals with a relatively high number of generations (100) is chosen in the optimal set. Only 50% of individuals in new generations are created using crossover, while the mutation rate of each gene is fixed at 5.9%. A highly nonlinear penalty function (with an exponent of 3) is used with a multiplication factor of 10, indicating stronger penalization of severely unfitted individuals or weaker penalization of barely unfitted individuals.

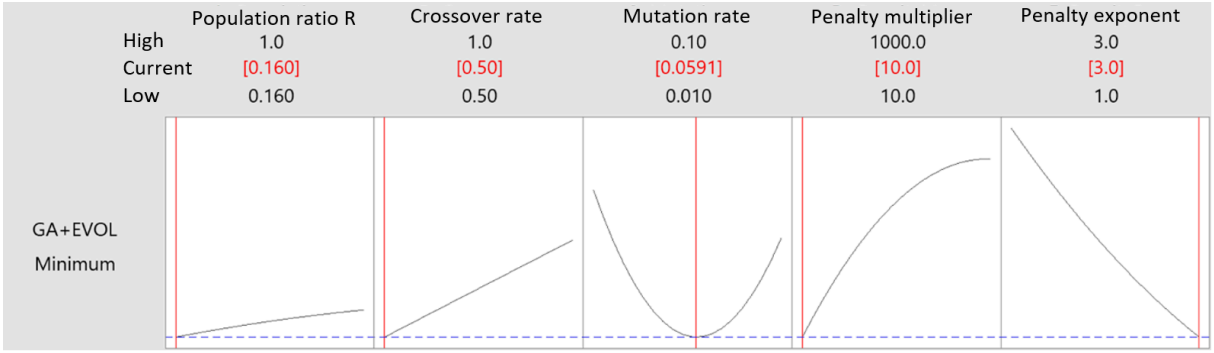


Figure 11. The optimized parameters for the hybrid algorithm (single-island case).

Once again, 10 repetitions of the new scenario are run in order to verify the performance of the algorithm. Indeed, the average of best-fitted individuals from every 10 runs is characterized by the lowest value of the objective function among all of the analyzed hybrid algorithm parameters, in accordance with RSM-based prediction. Even though the average objective function is relatively low, there is a visible spread in the results (min and max values are 0.257 and 0.356, respectively, with an average of 0.284), as shown in Figure 12.

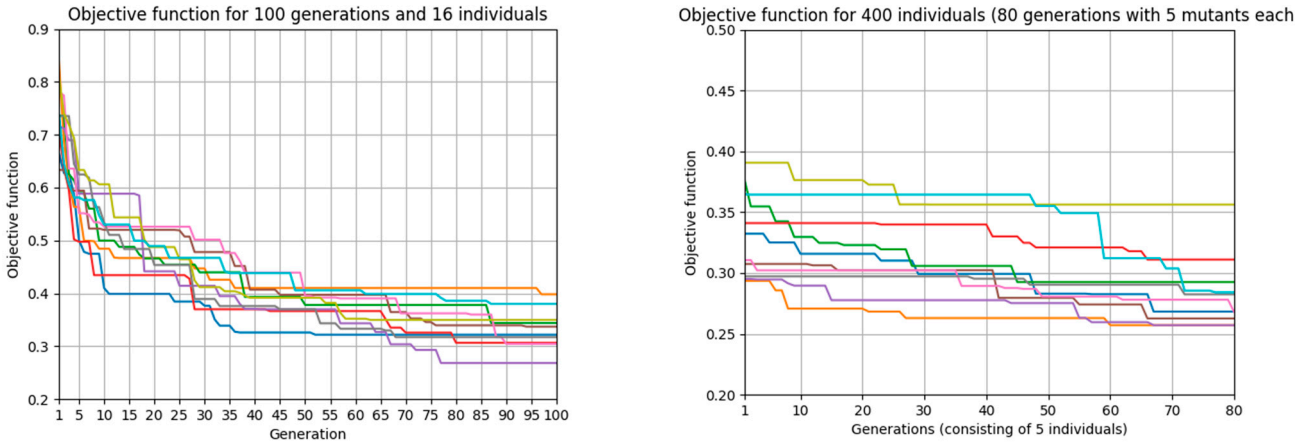


Figure 12. The convergence of the proposed hybrid optimization algorithm (10 runs)—single-population case.

3.2.2. Multi-Island Genetic Algorithm and Evolution Strategy Results

Once the genetic operators and penalty function parameters were fixed, the second study, this time on the multi-island version of the genetic algorithm, was conducted. Again, the analysis of the fitted full-quadratic response surface shows the dependencies between the factors and resultant objective function values (Figure 13).

A strongly nonlinear effect can be seen in all factors besides the first parameter—the number of autonomous populations (islands). On the other hand, this parameter has the strongest effect on the objective function value, despite being almost linear in its nature. Combining the results from the first and second study, one may conclude, that for this particular example, the total number of function calls is severely limited (in accordance with the given calculation time bounds); therefore, introducing multiple autonomous populations strongly limits the number of generations (or the number of individuals per each population). In effect, this diminishes the chance of converging to the optimum values, due to insufficient evolution of the population(s). Specifically, the desired evolution could be obtained only once the total number of generations or individuals (or both) is increased.

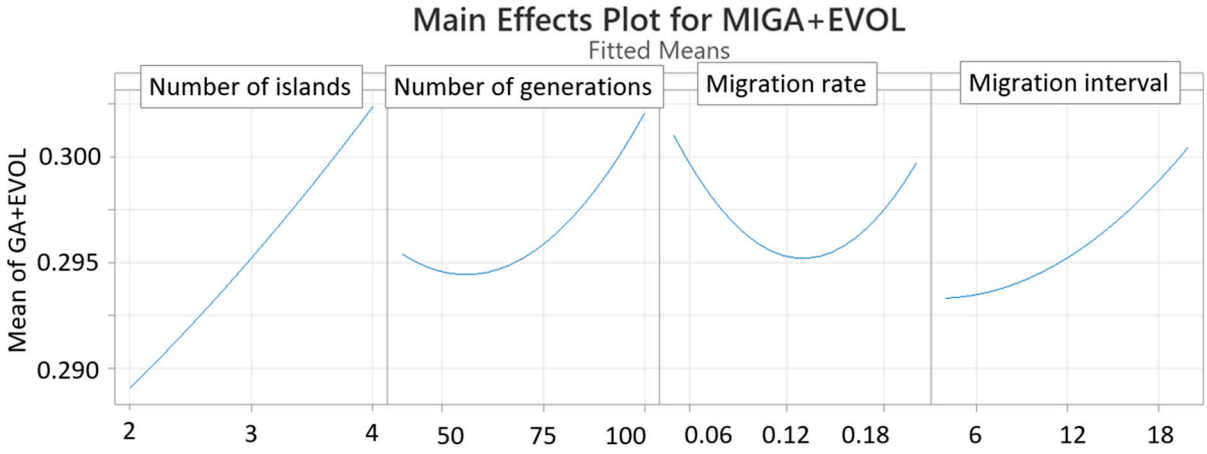


Figure 13. The main effect plot for the hybrid optimization using MIGA + EVOL approach (mean value of the fitted RSM shown in the Y axis).

Nevertheless, this study delivered multiple interesting observations regarding the mutual interactions between the studied factors (see Figure 14 for details). First, the most significant interaction (statistically) is the one between the number of autonomous populations (islands) and the migration rate. For a relatively small rate of migration (5%), increasing the number of islands helped in the attainment of a lower resultant objective function value. For higher values, i.e., 12.5% and 20%, more islands were an obstacle to the efficient minimization of the objective function.

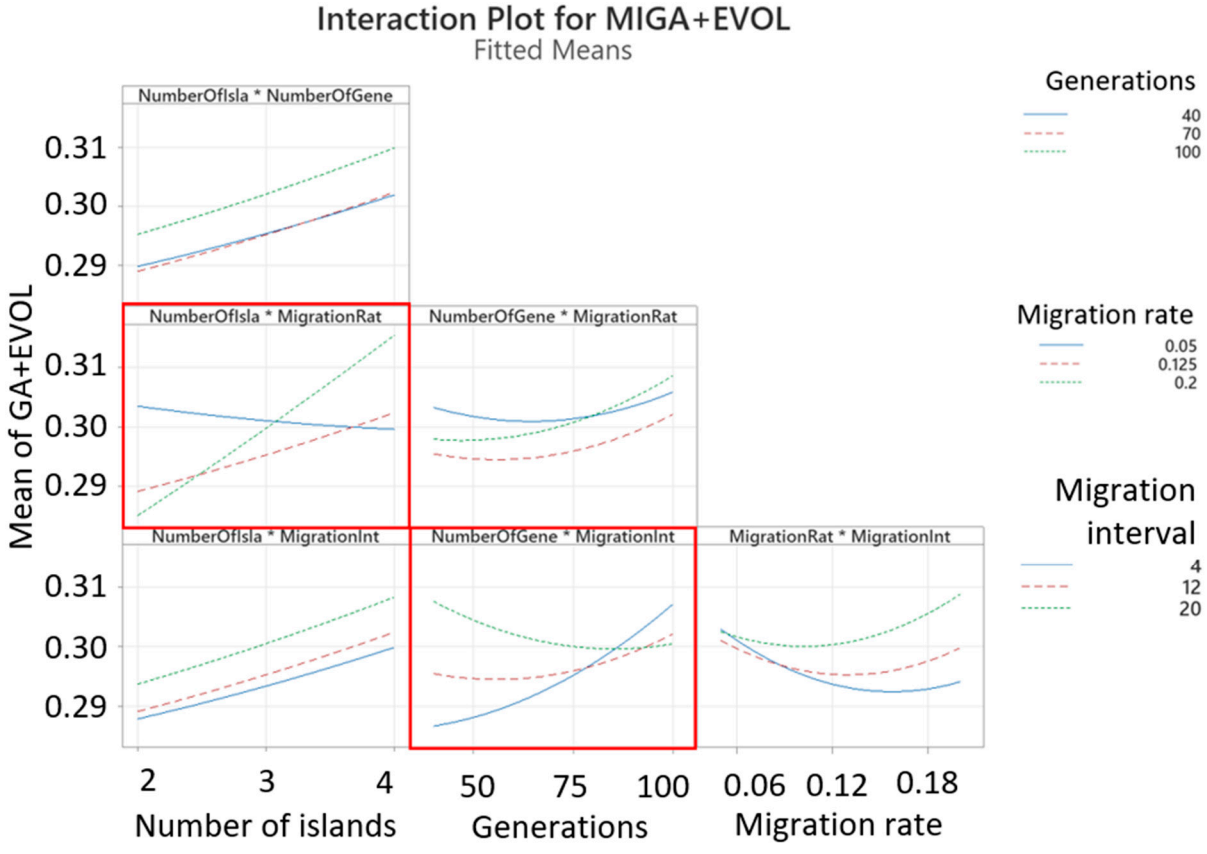


Figure 14. The interactions between considered factors for the MIGA + EVOL case.

The second most important interaction is the one between the number of generations and the interval of migration between the islands. It not only shows the strongly nonlinear

relationship between the factors, but also the opposite trend for extreme values of the parameters. For frequent migration (interval = 4), the population with a higher number of individuals and a lower number of generations is preferred. On the other hand, when the migration interval is increased to 20, a significantly higher number of generations is needed to minimize the objective function.

Once again, based on the fitted quadratic response surface, the authors adjusted the considered parameters to produce the lowest average objective function utilizing only a limited number of simulations. The results of this optimization are presented in Figure 15. The most efficient optimization is anticipated when high randomization is introduced to the population, i.e., there are only two islands, each with forty randomly generated individuals, and the migration occurs for 20% of the individuals per every four generations (with forty generations in total). This produces a very diverse population, as gene exchange is frequent and broad.

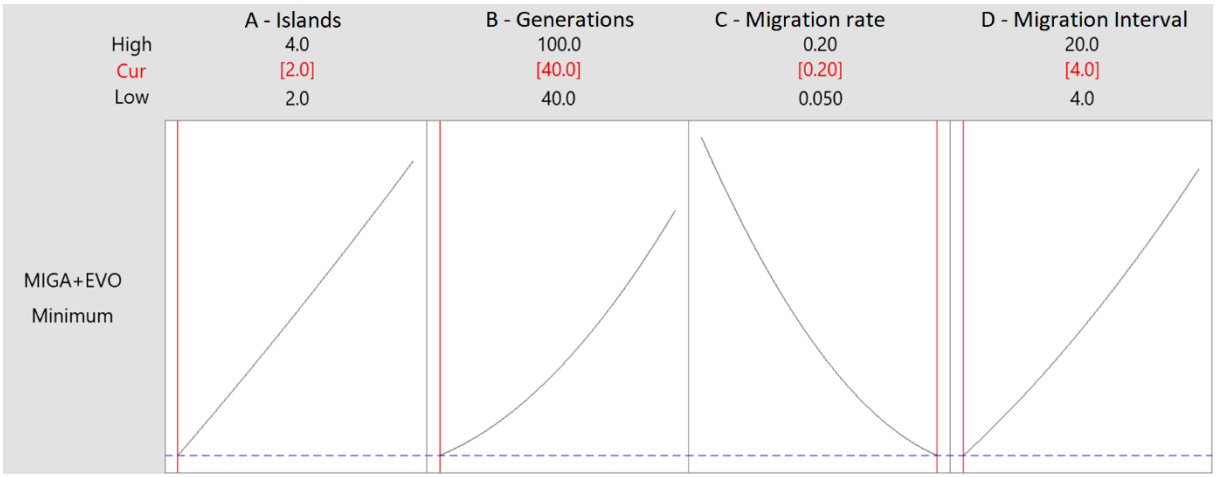


Figure 15. The optimized parameters for the hybrid algorithm (multi-island case).

Similar to the single-population version of the evolution algorithm study, the convergence of 10 runs is shown in Figure 16. This time, the average objective function is 0.285, with min and max values of 0.266 and 0.305, which in total yields a lower spread, compared to the previous study. As the average objective functions differ only by 0.001, both algorithms may be considered equally efficient within the considered scenario, with the multi-island approach having a more narrow span of the resultant best-fitted individuals.

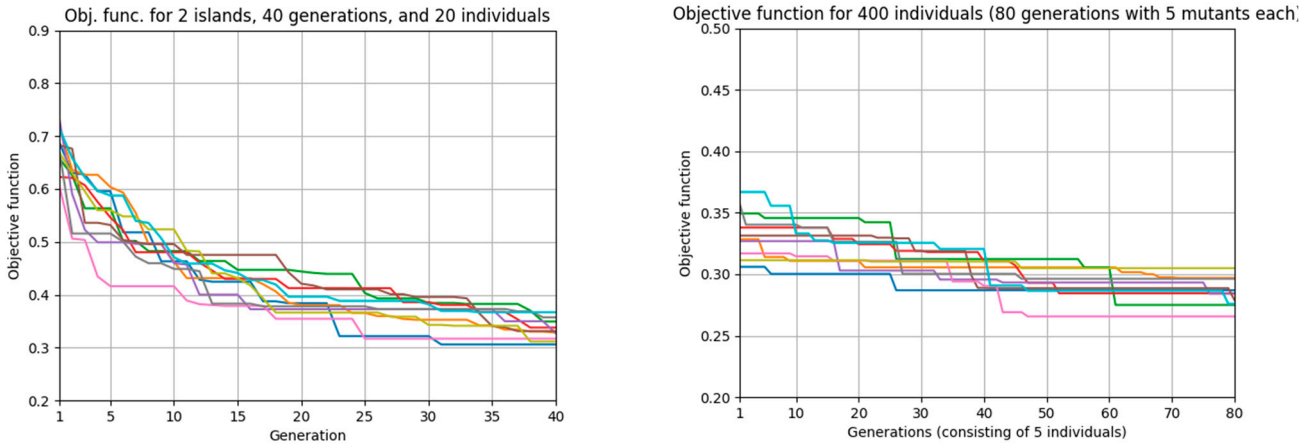


Figure 16. The convergence of the proposed hybrid optimization algorithm (10 runs)—multi-island case.

3.2.3. Results Comparison of the Hybrid Algorithm vs. Genetic Algorithm with the Same Number of the Total Function Calls

The proposed hybrid approach consists of two evolutionary algorithms which are called one after another. To justify this connection, which is rather uncommon, the authors decided to compare the proposed method with a single genetic algorithm, but with the increased number of allowed function calls—from 1600 to 2000 (i.e., equal to the total function calls used in the hybrid approach), to achieve the direct comparison between hybrid and purely genetic approach. The study was conducted for both—the single population and the multi-island version of the genetic algorithm, and similar to the previous examples, both settings were checked using 10 repetitions of the optimization process.

As the limit of function calls is increased to match the one from the hybrid approach, two cases are examined in the study—one, in which the total number of generations is increased by 20%, and the second, in which the total number of individuals per population is increased by 20%. The overview of the resultant average best individuals (among 10× repetitions) is shown in Table 4.

Table 4. Comparison of different strategies (using a total of 2000 function calls).

Strategy	Average Best Individual
Hybrid—single population	0.284
Hybrid—multi-island	0.285
GA, single population—more generations	0.326
GA, single population—more individuals	0.310
GA, multi-island—more generations	0.314
GA, multi-island—more individuals	0.320

The above results show that the hybrid approach outperformed the application of a single genetic algorithm, even though the same amount of objective function calls were used in all cases. Moreover, some insight into the nature of the GA can be observed, i.e., in a single population case, where the number of individuals is relatively small (16 individuals), increasing the number of generations (from 100 to 125) was not as helpful as the enlargement of the population size (from 16 to 20). On the other hand, in the multi-island case, adding more individuals (from 2×20 to 2×25) was less efficient than increasing the total number of generations (from 40 to 50).

The convergence of all four algorithms is shown in Figures 17 and 18.

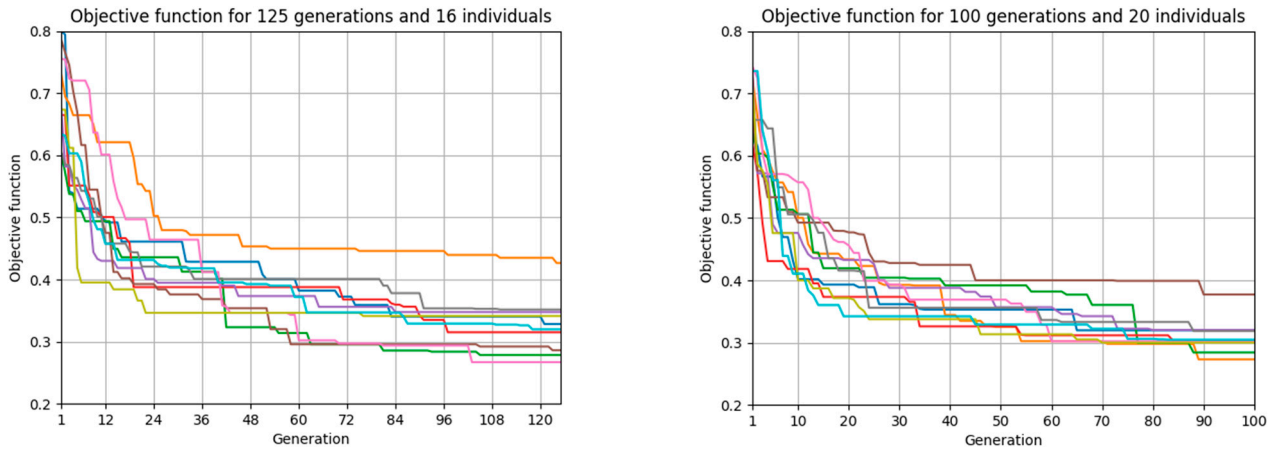


Figure 17. The convergence of the single population genetic algorithm with 2000 function calls.

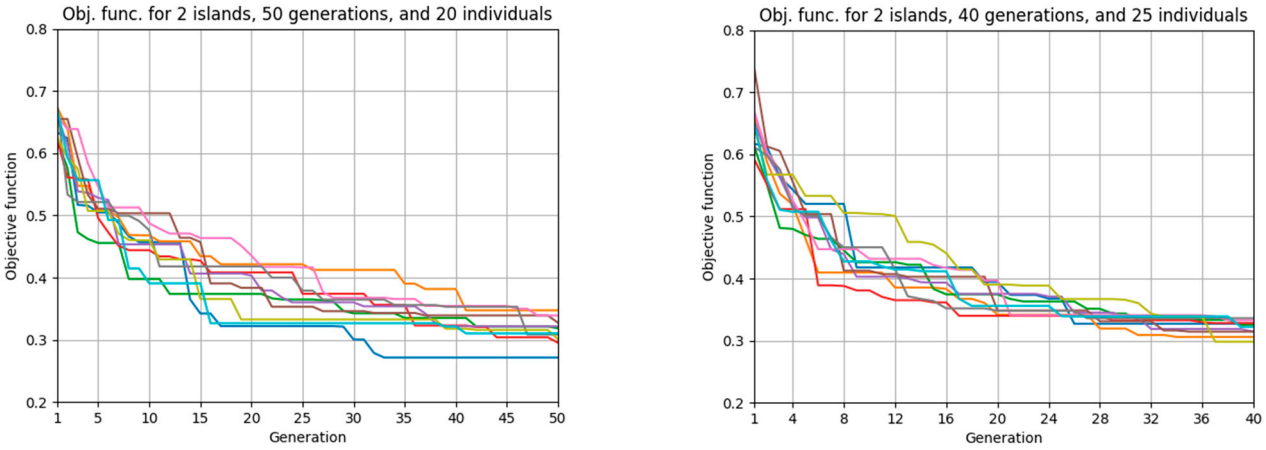


Figure 18. The convergence of the multi-island genetic algorithm with 2000 function calls.

3.3. Implementation of the Tuned Hybrid Optimization Algorithm Parameters to the FEM-Based Constrained Mass Minimization Problem

The last part of the study was the verification of the newly obtained optimization scenario in the FEM-based mass minimization problem with limit load constraint. Both of the tuned hybrid algorithms were examined, as the actual simulation-based coarse response may differ from the smooth metamodel-based study. Additionally, for comparison purposes, the authors ran the optimization with parameters corresponding to the lowest efficiency of the hybrid approach, to determine whether this trend will be correctly captured by the FEM-based optimization.

The convergence of all of the analyzed cases is shown below. To minimize the effect of randomness, each FEM-based study was repeated three times. As all algorithms have a different number of individuals and generations, the only common feature is the number of function calls (individuals calculated); therefore, it was chosen as the X-scale in Figure 19.

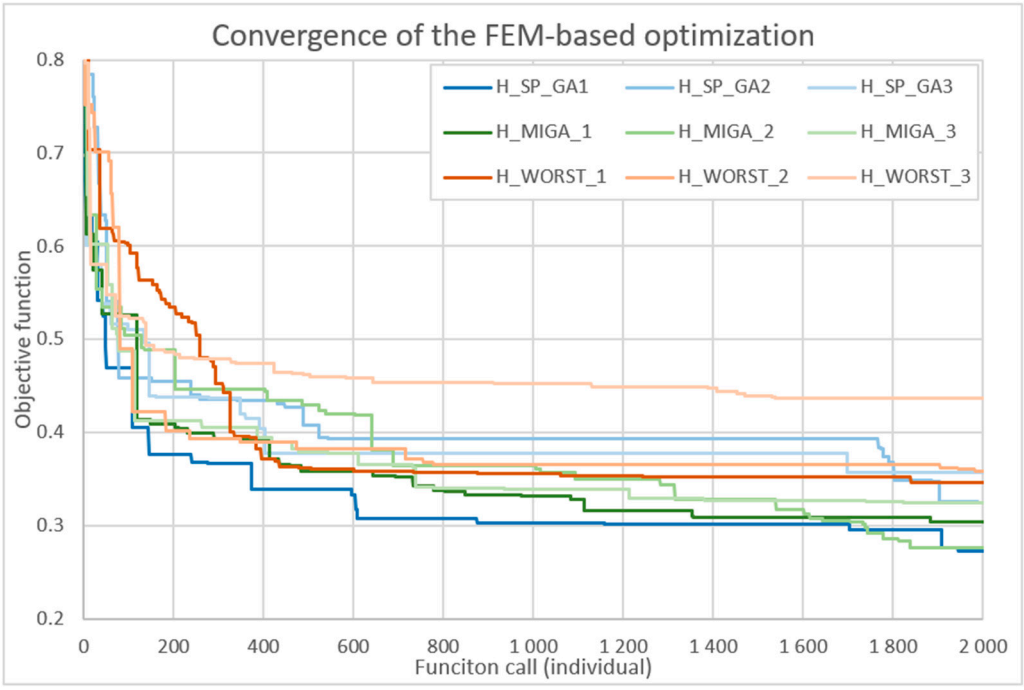


Figure 19. The convergence of the FEM-based optimization: H_SP_GA—hybrid approach with a single population genetic algorithm, H_MI_GA—hybrid approach with a multi-island genetic algorithm, and H_WORST—hybrid approach with properties corresponding to the least-performing algorithm in accordance with the metamodel-based study.

The resultant designs of the bracket being optimized differ in terms of the final shape and mass. The average values of the best individuals after each stage of the hybrid optimization are listed in Table 5.

Table 5. Comparison of the average best individual objective function values obtained from different algorithm parameters using FEM-based optimization.

Average Best Individual	Single Population GA + EVOL	Multi-Island GA + EVOL	Worst Case GA + EVOL
After GA	0.358	0.318	0.385
After GA + EVOL	0.318	0.301	0.380

The resultant shape of the bracket cannot be averaged; therefore, the best designs from the H_SP_GA and H_MI_GA are shown together (Figure 20) with one of the resultant H_WORST designs. It can be seen, that for the two first cases, the bracket design is similar (thickness, general contour), with some differences observed mostly in the lower section. On the other hand, the design from the H_WORST algorithm (which is the least performing) is overly heavy, with a very thick lower part connected with the upper region via an hourglass-shaped contour, which from a strength perspective should be avoided.

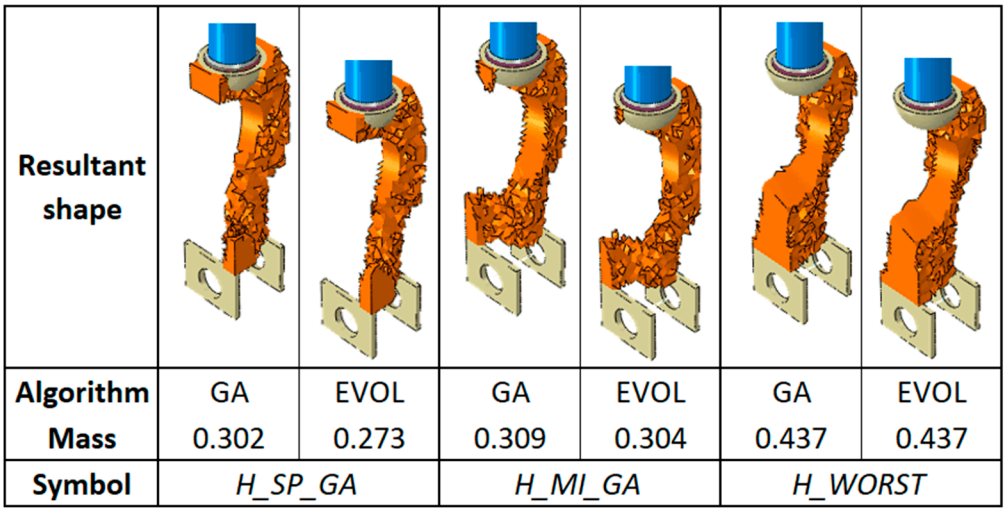


Figure 20. The comparison of the final design mass and shape: H_SP_GA—hybrid approach with a single population genetic algorithm, H_MI_GA—hybrid approach with a multi-island genetic algorithm, and H_WORST—hybrid approach with properties corresponding to the least-performing algorithm in accordance with the metamodel-based study.

4. Conclusions

The conducted study has shown, that the usage of metamodels may greatly increase the performance of the optimization algorithms. The authors were able to tune the optimization parameters to increase the efficiency of the proposed algorithms by respectively 12% and 5% for single- and multi-island versions of the hybrid algorithm (comparing the resultant optimal average objective function to the one obtained as an average for all of the analyzed runs). Moreover, the proposed approach enables reducing the total number of parameter combinations to study, as the experiments are divided into single- and multi-population implementation of the genetic algorithm separately (however, sequentially).

To conduct this study, one does not necessarily require an excessive sampling plan for the metamodel training phase. The authors have shown, that even for a significant number of variables, the performance of the metamodels trained on smaller data sets is mildly decreased (R^2 drop of 10% for the response F_{LIMIT}) or almost the same (R^2 reduced by 2% for response Mass) compared to the four times bigger sampling plan

used in the considered work. Nonetheless, the performance of the metamodel is strongly dependent on the underlying nature of the problem. The authors presented the nonlinear buckling example, which included both geometrical and material nonlinearities, making the constraint function extremely sensitive to the design variables.

Compared to other studies described in the literature, this one handles both the evolutionary operators as well as penalty-related ones (as the constrained-optimization problem is considered). This can lead to surprising conclusions, such as the fact that a proper penalty function may have a more significant impact on the optimization efficiency, than the number of individuals or generations of the genetic algorithm. However, this observation may be problem-related, as the authors considered a fixed amount of objective function calls, which may not guarantee the attainment of the global optimum. Yet, in real-world engineering optimization problems, one cannot find a single algorithm that will assure this global optimum each time. Nonetheless, the presented method of parameters tuning is universal in its nature and may be adopted in almost all types of optimization algorithms.

In stochastic processes, such as in genetic algorithms and evolution strategies, it is beneficial to increase the number of algorithm repetitive runs, which is relatively costless when working on quick-to-run metamodels rather than the actual FEM-based or other time-consuming simulations. In this case, the noise associated with the natural spread of evolutionary methods is mitigated, allowing for drawing valid conclusions regarding the algorithm optimization process.

Finally, tests performed on the actual FEM-based optimization with the proposed hybrid approach show, that the conclusions withdrawn from the metamodel-based study are valid and they indeed helped in increasing the efficiency of the optimization process. The same observations have been made regarding the level of spread in the single- and multi-population versions of the genetic algorithm in both the FEM and metamodel-based optimizations, i.e., the MIGA approach yielded a more narrow distribution of best individuals compared to the single-population version of the algorithm.

This study may be extended to verify the performance of adaptive genetic operators, such as [44], or nonconstant migration properties, such as [45]. In addition, usage of elemental response (such as, e.g., Huber−Mises−Hencky equivalent stress) may seem as a useful heuristic to steer the optimization process. Moreover, the trained metamodels may be useful for finding areas of a low objective function, where local-search algorithms based on the actual FEM simulation may be initiated from—this could resolve the issue of finding the starting point for the deterministic optimizations. Generally, the further improvement of the optimization capabilities for the presented industrial-based problem is the main goal of the authors in subsequent work. Nonetheless, the presented method for parameter tuning of hybrid optimization is universal and may be adapted to numerous fields of engineering.

Author Contributions: Conceptualization, W.K. and P.S.; methodology, W.K. and P.S.; software, P.S.; writing—original draft preparation, P.S.; writing—review and editing, W.K. and P.S. All authors have read and agreed to the published version of the manuscript.

Funding: The research was co-financed under grant no. DWD/3/7/2019 supported by the Ministry of Science and Higher Education in Poland and a research subsidy of the Mechanical Engineering Faculty, Silesian University of Technology, 2023.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sigmund, O. A 99 line topology optimization code written in Matlab. *Struct. Multidiscip. Optim.* **2001**, *21*, 120–127. [CrossRef]
2. Lindgaard, E.; Dahl, J. On compliance and buckling objective functions in topology optimization of snap-through problems. *Struct. Multidiscip. Optim.* **2013**, *47*, 409–421. [CrossRef]

3. Rad, M.M.; Habashneh, M.; Lógó, J. Elasto-Plastic limit analysis of reliability based geometrically nonlinear bi-directional evolutionary topology optimization. *Structures* **2021**, *34*, 1720–1733. [CrossRef]
4. Allaire, G.; Dapogny, C.; Jouve, F. Shape and topology optimization. In *Handbook of Numerical Analysis*; Elsevier: Amsterdam, The Netherlands, 2021; pp. 1–132. [CrossRef]
5. Mollik, T.; Geng, Y.; Shougat, M.R.E.U.; Fitzgerald, T.; Perkins, E. Genetic algorithm shape optimization to manipulate the nonlinear response of a clamped-clamped beam. *Heliyon* **2022**, *8*, e11833. [CrossRef]
6. Kieszek, R.; Kachel, S.; Kozakiewicz, A. Modification of Genetic Algorithm Based on Extinction Events and Migration. *Appl. Sci.* **2023**, *13*, 5584. [CrossRef]
7. Zhang, J.J.; Xu, L.W.; Gao, R.Z. Multi-island Genetic Algorithm Opetimization of Suspension System. *TELKOMNIKA* **2012**, *10*, 1685–1691. [CrossRef]
8. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126. [CrossRef]
9. Chen, Z.; Wang, Z.; Zhang, Q.; Liu, Z.; Pan, M.; Chen, D.; Xu, Y. Dynamic Compensation of a Fluxgate Magnetometer Based on a Hybrid Optimizing Algorithm. *Appl. Sci.* **2023**, *13*, 2830. [CrossRef]
10. Tarek, E.M.; Hopgood, A.; Nolle, L.; Battersby, A. Hybrid Genetic Algorithms: A Review. *Eng. Lett.* **2006**, *13*, 1–14. Available online: https://www.engineeringletters.com/issues_v13/issue_2/EL_13_2_11.pdf (accessed on 1 May 2023).
11. Pasandideh, S.H.R.; Niaki, S.T.A.; Tokhmehchi, N. A parameter-tuned genetic algorithm to optimize two-echelon continuous review inventory systems. *Expert Syst. Appl.* **2011**, *38*, 11708–11714. [CrossRef]
12. Fattahi, P.; Hajipour, V.; Hajiloo, S. A multi-objective parameter-tuned soft computing-based algorithm to optimize competitive congested location-pricing problem within multi-type service. *Array* **2021**, *10*, 100062. [CrossRef]
13. Wang, G.G.; Shan, S. Review of Metamodeling Techniques in Support of Engineering Design Optimization. *J. Mech. Des.* **2007**, *129*, 370–380. [CrossRef]
14. Bhosekar, A.; Ierapetritou, M. Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Comput. Chem. Eng.* **2018**, *108*, 250–267. [CrossRef]
15. Soares Do Amaral, J.V.; Montevechi, J.A.B.; Miranda, R.D.C.; Junior, W.T.D.S. Metamodel-based simulation optimization: A systematic literature review. *Simul. Model. Pract. Theory* **2022**, *114*, 102403. [CrossRef]
16. He, C.; Zhang, Y.; Gong, D.; Ji, X. A review of surrogate-assisted evolutionary algorithms for expensive optimization problems. *Expert Syst. Appl.* **2023**, *217*, 119495. [CrossRef]
17. Kieszek, R.; Kozakiewicz, A.; Rogólski, R. Optimization of a Jet Engine Compressor Disc with Application of Artificial Neural Networks for Calculations Related to Time and Mass Criteria. *Adv. Sci. Technol. Res. J.* **2021**, *15*, 208–218. [CrossRef]
18. Baklouti, A.; Dammak, K.; El Hami, A. Uncertainty Analysis Based on Kriging Meta-Model for Acoustic-Structural Problems. *Appl. Sci.* **2022**, *12*, 1503. [CrossRef]
19. Sbayti, M.; Ghiotti, A.; Bahloul, R.; BelhadjSalah, H.; Bruschi, S. Effective strategies of metamodeling and optimization of hot incremental sheet forming process of Ti6Al4V artificial hip joint component. *J. Comput. Sci.* **2022**, *60*, 101595. [CrossRef]
20. Pourmohammadi, P.; Saif, A. Robust metamodel-based simulation-optimization approaches for designing hybrid renewable energy systems. *Appl. Energy* **2023**, *341*, 121132. [CrossRef]
21. Sebastjan, P.; Kuś, W. Optimization of material distribution for forged automotive components using hybrid optimization techniques. *Comput. Methods Mater. Sci.* **2021**, *21*, 63–74. [CrossRef]
22. *ABAQUS Analysis User's Manual*; Version 2022; Dassault Systemes Simulia, Inc.: Johnston, RI, USA, 2022.
23. Jin, R.; Chen, W.; Sudjianto, A. An efficient algorithm for constructing optimal design of computer experiments. *J. Stat. Plan. Inference* **2005**, *134*, 268–287. [CrossRef]
24. Morris, M.D.; Mitchell, T.J. Exploratory designs for computational experiments. *J. Stat. Plan. Inference.* **1995**, *43*, 381–402. [CrossRef]
25. Chowdhury, M.; Alouani, A.; Hossain, F. Comparison of ordinary kriging and artificial neural network for spatial mapping of arsenic contamination of groundwater. *Stoch. Environ. Res. Risk Assess.* **2010**, *24*, 1–7. [CrossRef]
26. McCulloch, W.S.; Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bull. Math. Biophys.* **1943**, *5*, 115–133. [CrossRef]
27. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
28. Al-Mudhafar, W.J.; Sepehrnoori, K. Designed Simulations for Optimization of Hydraulic Fracture Design and Production Well Constraints in Shale Gas Reservoirs with Reduced-Physics Metamodeling. In Proceedings of the SPE Europec Featured at 80th EAGE Conference and Exhibition 2018, Copenhagen, Denmark, 11–14 June 2018. [CrossRef]
29. Song, C.Y. A Study on Learning Parameters in Application of Radial Basis Function Neural Network Model to Rotor Blade Design Approximation. *Appl. Sci.* **2021**, *11*, 6133. [CrossRef]
30. Hinton, G.E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors. 2012. Available online: https://arxiv.org/abs/1207.0580 (accessed on 17 April 2023).
31. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.

32. Le, Q.V.; Ngiam, J.; Coates, A.; Lahiri, A.; Prochnow, B.; Ng, A.Y. On Optimization Methods for Deep Learning. In Proceedings of the 28th International Conference on International Conference on Machine Learning, Bellevue, WA, USA, 28 June–2 July 2011; Available online: https://cs.stanford.edu/~acoates/papers/LeNgiCoaLahProNg11.pdf (accessed on 17 April 2023).
33. Kumar, S.D.V.; Kai, M.L.Y.; Arumugam, T.; Karuppanan, S. A Review of Finite Element Analysis and Artificial Neural Networks as Failure Pressure Prediction Tools for Corroded Pipelines. *Materials* **2021**, *14*, 6135. [CrossRef]
34. Krige, D.G. A statistical Approach to Some Mine Valuations and Allied Problems at the Witwatersrand. Master's Thesis, University of Witwatersrand, Johannesburg, South Africa, 1951.
35. Roux, E.; Bouchard, P.-O. Kriging metamodel global optimization of clinching joining processes accounting for ductile damage. *J. Mater. Process. Technol.* **2013**, *213*, 1038–1047. [CrossRef]
36. Raponi, E.; Bujny, M.; Olhofer, M.; Aulig, N.; Boria, S.; Duddeck, F. Kriging-assisted topology optimization of crash structures. *Comput. Methods Appl. Mech. Eng.* **2019**, *348*, 730–752. [CrossRef]
37. Couckuyt, I.; Forrester, A.; Gorissen, D.; Turck, F.D.; Dhaene, T. Blind Kriging: Implementation and performance analysis. *Adv. Eng. Softw.* **2012**, *49*, 1–13. [CrossRef]
38. Hwang, G.-H.; Jang, W.-T. An Adaptive Evolutionary Algorithm Combining Evolution Strategy and Genetic Algorithm (Application of Fuzzy Power System Stabilizer). In *Advances in Evolutionary Algorithms*; Kosiński, W., Ed.; InTech.: Chatham, UK, 2008. [CrossRef]
39. Whitley, D.; Rana, S.; Heckendorn, R.B. The Island Model Genetic Algorithm: On Separability, Population Size and Convergence. *J. Comput. Inf. Technol.* **1998**, *1*, 33–47.
40. Beyer, H.-G.; Schwefel, H.-P. Evolution strategies—A comprehensive introduction. *Nat. Comput.* **2002**, *1*, 3–52. [CrossRef]
41. Mosayebi, M.; Sodhi, M. Tuning genetic algorithm parameters using design of experiments. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, ACM, Cancún, Mexico, 8–12 July 2020; pp. 1937–1944.
42. Mobin, M.; Mousavi, S.M.; Komaki, M.; Tavana, M. A hybrid desirability function approach for tuning parameters in evolutionary optimization algorithms. *Measurement* **2018**, *114*, 417–427. [CrossRef]
43. Minitab. 2021. Available online: https://www.minitab.com/ (accessed on 17 April 2023).
44. Hassanat, A.; Almohammadi, K.; Alkafaween, E.; Abunawas, E.; Hammouri, A.; Prasath, V.B.S. Choosing Mutation and Crossover Ratios for Genetic Algorithms—A Review with a New Dynamic Approach. *Information* **2019**, *10*, 390. [CrossRef]
45. Li, C.; Wu, J. Subpopulation Diversity Based Selecting Migration Moment in Distributed Evolutionary Algorithms. 2017. Available online: http://arxiv.org/abs/1701.01271 (accessed on 17 April 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.