ONLINE FIRST July 3, 2026

Computer Assisted Methods in Engineering and Science
***, https://doi.org/10.24423/cames.2026.1971

Metamodel Selection and Hyperparameter Tuning for Hybrid Optimization with a Quantum-Inspired Evolutionary Algorithm in Multiple Drop-Test Simulation

Adam Rurański, Wacław Kuś*

Department of Computational Mechanics and Engineering,
Faculty of Mechanical Engineering, Silesian University of Technology, Gliwice, Poland

* *Corresponding Author: waclaw.kus@polsl.pl*

This study introduces a hybrid optimization framework for the multi-drop testing of a lithium-ion battery enclosure. The framework integrates a quantum-inspired evolutionary algorithm (QEA) with surrogate modeling techniques. The contribution of the present study lies in metamodel selection, hyperparameter tuning, and the evaluation of two hybrid integration strategies built on a previously published QEA framework. Three types of metamodels were applied, artificial neural networks (ANNs), Kriging, and polynomial regression (PNR), using datasets generated via Latin hypercube sampling and from prior QEA iterations. Hyperparameter tuning methods constitute the main part of the study. Two fitness functions were analyzed, including a logarithmically scaled variant designed to compress the output range for damaged cases and enhance classification accuracy near the damage/no-damage boundary. A dual-model strategy was employed for the ANN, with model switching determined by a plastic strain threshold. Across the tested datasets, the ANN frequently identified low-objective individuals, while PNR occasionally exhibited instability. Two hybrid schemes were implemented: HYBRID1 yielded the lowest objective values in the reported runs, whereas HYBRID2 reduced runtime and showed a steadier optimization course, at a slight cost to the best obtained objective value. Within the analyzed QEA-based workflow, the combination of QEA, ANN, and the second objective fitness function (FF2) reduced finite element method (FEM) computational time by an order of magnitude while maintaining decision-making effectiveness. These results support the proposed hybridization as a practical improvement of the existing QEA workflow for the studied multi-drop test problem.

Keywords: surrogate modeling, metamodel selection, hyperparameter tuning, artificial neural network, Kriging, polynomial regression, quantum-inspired evolutionary algorithm, multi-drop test, battery housing.

1. Introduction

Nowadays, highly computationally demanding tasks occur in routine engineering practice. The standard approach to perform analyses using only FEM

and optimization techniques, particularly in the optimization process, is no longer sufficient. It is becoming increasingly common to combine other techniques to improve efficiency and reduce computational costs [1, 6]. Optimization-based methods have already been successfully applied to various engineering problems, including structural applications, which further motivates the development of more efficient optimization workflows [7, 9]. In recent years, artificial intelligence (AI) tools have started to be implemented even in commercial industrial software such as the 3DEXPERIENCE platform [10], Ansys [11], and Altair [12]. However, these tools cannot address all existing industrial needs. Certain cases and tasks still require not a general but a specific, individual approach, often based on existing tools but with the addition of new scripts, interfaces or problem-solving approaches [13]. In addition to the parametric optimization tools that are already implemented, there remains potential to reduce computational costs (defined as the wall-clock time of computations) by predicting objective function values based on previously computed cases stored in databases and/or supported by metamodels. Such surrogate-based strategies have been shown to effectively accelerate repetitive simulations by replacing full high-fidelity models with appropriately trained reduced-order or metamodel representations [14, 15].

An example of such a problem is the optimization of a battery housing – a component exposed to multi-drop tests according to the standard code [16]. Many devices used in industry, as well as home appliances, are at risk of being dropped, prompting industry to conduct tests to ensure the intended and reliable safety performance. Devices subjected to multiple drops include battery-powered tools, such as electronic equipment or garden tools. It is also important to note that batteries present an elevated risk of fire and/or explosion. FEM tools currently offer full capabilities to perform drop-test simulations and can be extended though open application programming interfaces (APIs) and scripting support. The automation of multi-drop test loops and customized optimization tools [16] enables the solution of the optimization process. However, the computational costs are high and it is important to reduce them to enable the application of these methods in industrial engineering tasks in daily practice.

In this study, the authors propose a method based on previously published works and present an improved approach that significantly reduces computation time. To reduce computational times, the authors propose supporting the optimization with metamodeling techniques, including ANNs, the Kriging method, and PNR. All of these metamodeling techniques are compared and evaluated for prediction efficiency. Ultimately, the paper evaluates surrogate-assisted extensions of the previously established QEA workflow and compares two QEA-based hybrid approaches.

To provide a clear structure of this work, Sec. 2 describes in detail the problem formulation, objective functions, database preparation using LHS and QEA approaches, prediction strategies, hyperparameter tuning procedures, and the implementation of the two proposed hybrid strategies (HYBRID1 and HYBRID2). Section 3 presents a numerical example of a battery housing drop-test model, along with database structures of various sizes and the process of tuning ANN, Kriging, and PNR metamodels. Optimization processes based on predictions from different metamodels are compared, and the most suitable metamodel is selected for final implementation in the HYBRID optimization approach. The results of the optimization obtained with the ANN metamodel are presented and summarized in Sec. 4. It should be noted that the primary goal of this study is to investigate the behavior and efficiency of hybrid optimization strategies combined with different metamodeling techniques within a computationally demanding problem. Therefore, a single well-defined engineering case was selected to enable a controlled and detailed analysis. The generalization of the proposed approach to other classes of problems is considered as a direction for future work.

2. Materials and methods

2.1. Optimization problem formulation

We consider an object that is exposed to a multi-drop test scenario. The multi-drop test is an experimental procedure involving repeated, controlled drops of a tested object from a specified height onto a defined surface to evaluate its mechanical resistance and durability. Between successive drops, the orientation of the object is altered to reproduce various possible real-world impact scenarios (e.g., corner, edge, or flat-surface impact) (Fig. 1). The number of drops, drop height, and type of impact surface are defined according to relevant standards or design specifications. This method enables the assessment of structural degradation due to repeated impact loads and, for critical components such as battery modules, allows verification of operational safety. In this

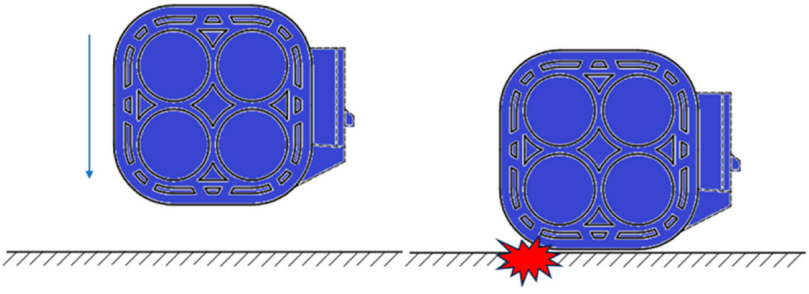


Fig. 1. Object subjected to a multi-drop test.

example, we analyze the housing of a lithium-ion battery. Similarly, as shown in the paper, the approach can be used for components and constructions under dynamic loading.

The aim of the optimization in this case is to design a structure that, when exposed to the multi-drop test, accumulates as little damage as possible. The following objective function was formulated:

$$\min_{P} J_0\left(\mathbf{P}\right). \tag{1}$$

We seek the values of the design vector $\mathbf{P}$ for which the function $J_0\left(\mathbf{P}\right)$ reaches its minimum value. The design vector $\mathbf{P}$ is defined as:

$$\mathbf{P} = [P_1, P_2, ..., P_n], \tag{2}$$

and the following constraints are given:

$$\mathrm{C} = \{\mathbf{P} : \forall \mathrm{i} \in \{1, ..., n\}\, P_{i\,\min} \leq P_i \leq P_{i\,\max}\}, \tag{3}$$

where $P_{i\,\min}$ and $P_{i\,\max}$ are minimum and maximum values of i-th design variable, respectively. In the present study, the optimization problem was intentionally formulated as damage minimization within a fixed feasible design window. No explicit mass-related term was introduced, because the aim was to assess the behavior of the surrogate-assisted QEA workflow for a previously bounded engineering design space. In the considered application, the housing mass itself was not treated as a critical design driver, while structural integrity under repeated drops remained the primary criterion.

Two objective (fitness) functions were used and compared. The first one, called FF1, is defined as follows:

$$J_0\left(\mathbf{P}\right) = \mathrm{V}_{\mathrm{del}}\left(\mathbf{P}\right) + \varepsilon_{p\_\max}\left(\mathbf{P}\right), \tag{4}$$

and the second, FF2, is defined as:

$$J_0\left(\mathbf{P}\right) = \begin{cases} \varepsilon_{p\_\max}\left(\mathbf{P}\right), & \text{if} \quad \mathrm{V}_{\mathrm{del}}\left(\mathbf{P}\right) = 0, \\ a \cdot \log\left(\mathrm{V}_{\mathrm{del}}\left(\mathbf{P}\right) + 1\right) + b, & \text{if} \quad \mathrm{V}_{\mathrm{del}}\left(\mathbf{P}\right) \geq 0, \end{cases} \tag{5}$$

where a and b are coefficients introduced later during the parameter selection step (Subsec. 3.1) for a uniform distribution of fitness function values. $\mathrm{V}_{\mathrm{del}}\left(\mathbf{P}\right)$ is the volume of deleted finite elements due to damage, and $\varepsilon_{p\_\max}\left(\mathbf{P}\right)$ is the maximum equivalent plastic strain in the structure during the tests. FF1, defined by Eq. (4), is an objective function in which the optimization goal is defined as the algebraic sum of the volume of deleted elements $\mathrm{V}_{\mathrm{del}}\left(\mathbf{P}\right)$ and the maximum plastic strain $\varepsilon_{p\_\max}\left(\mathbf{P}\right)$. In this case, the range of objective function values

is expected to be relatively large when a high number of elements is removed due to damage, and very small for cases with a low number of damaged elements. For this reason, FF2, described by Eq. (5), was proposed, where a lower range of objective function values is expected due to the logarithmic function applied to the term depending on the volume of damaged elements.

The objective function values were obtained from an explicit FEM model of the battery housing subjected to the prescribed sequence of drop events. The model was parameterized by the design vector $\mathbf{P}$, and for each candidate design the same multi-drop loading scenario was simulated. Each objective-function evaluation corresponded to the full prescribed drop sequence for a given design. The structural state after each impact was transferred to the subsequent drop, and the reported values were extracted after completion of the entire sequence. The quantities extracted from the FEM analysis were the maximum equivalent plastic strain $\varepsilon_{p\_\max}(\mathbf{P})$ and the deleted-element volume $V_{\mathrm{del}}(\mathbf{P})$, which were then used to calculate the fitness functions FF1 and FF2. Damage was represented by damage initiation followed by damage evolution with stiffness degradation and element deletion. The FEM calculations were embedded in an automated script that evaluated consecutive design vectors and collected the quantities required for the objective function, as shown in Fig. 2.

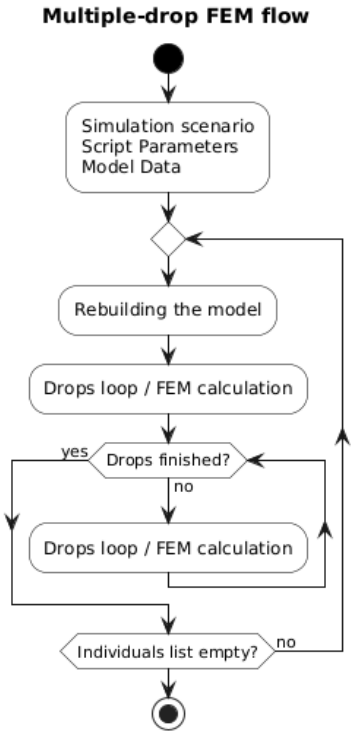


Fig. 2. Scheme for obtaining objective function values from FEM.

Cumulative damage is defined as a combination of the plastic strain calculated in Abaqus $\varepsilon_{p\_\max}(\mathbf{P})$ and the volume of deleted elements $\mathrm{V}_{\mathrm{del}}(\mathbf{P})$. The deleted volume is calculated on the basis of the sum of all removed elements (when a regular mesh is used). During the FEM analysis, elements are removed from the model on the basis of a damage initiation model [17] and a constitutive model of damage evolution. After the onset of damage, determined by the damage initiation model, the damage evolution process begins to reduce the stiffness of the element. Once the element loses stiffness, it is removed from the further steps of simulation.

2.2. Database of solutions and design of experiment

The results of multiple analyses mentioned above are collected in a database. Each row contains the values of the vector $\mathbf{P}$, and the remaining columns contain analysis results such as the number of deleted elements, the volume of deleted elements, and the maximum equivalent plastic strain. For comparison purposes, two types of databases were evaluated. The first one is based on a design vector obtained with the use of design of experiments (DOE) techniques. In our case, all design vectors were generated using LHS [18] with the CenteredMaxMin option. The LHS method with the centered-maximin criterion combines two strategies to improve the quality of sample distribution. Each sample is placed at the center of its stratified interval (centered), ensuring uniform coverage along individual dimensions. Simultaneously, the configuration is optimized to maximize the minimum Euclidean distance between any pair of points (maximin), which improves the overall space-filling properties. This approach results in well-distributed, low-correlation sampling designs, which are particularly suitable for surrogate modeling and DOE in high-dimensional spaces [19].

A few databases were created. A full, largest database was generated, and several smaller databases were created to compare the prediction quality of metamodels. The design vectors in all databases were generated separately. To avoid high computational costs, points from each of the smaller databases corresponding to the nearest points in the larger dataset were identified. These selected points were then transferred to a new, smaller dataset, as shown in Fig. 3.

It is important to note that the distance between points does not have to be calculated in real (three-dimensional) space. The distance can be computed using Eq. (6), even in a higher-dimensional space beyond the physical domain. This was done by applying the Euclidean distance between points in an n-dimensional space

$$d(a,b) = \sqrt{\sum_{i=1}^{m} (a_i - b_i)^2}, \tag{6}$$

where a_i and b_i are the values of the variables for two different points, and m is the number of dimensions. This approach enabled a significant reduction of computational time while maintaining a uniform distribution of the samples.

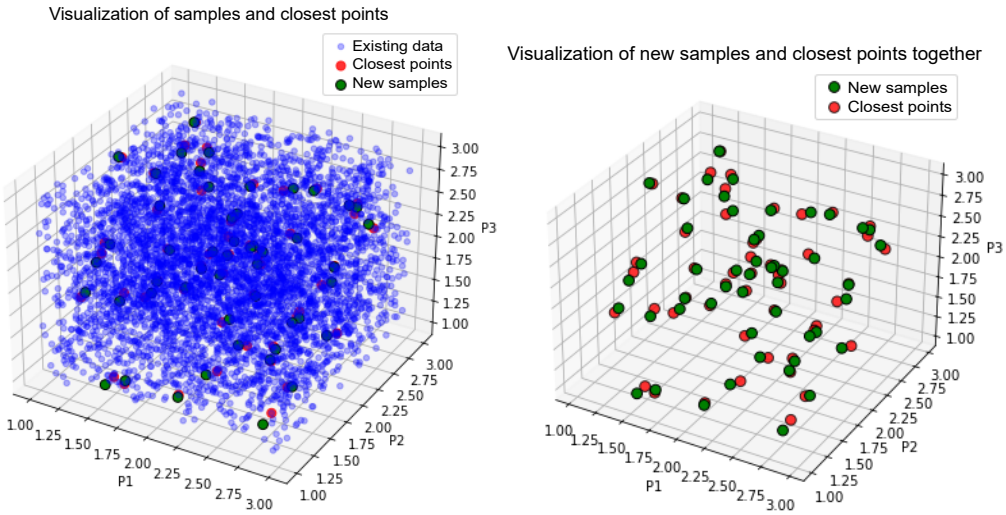


FIG. 3. Mapping reduced database samples to their closest points in the full database.

Another database was based on design vectors selected from previous runs of optimization using QEA [16]. Previously collected data from multiple runs of the optimization task provided a sufficient amount of data. The data in this database are not as evenly distributed as in the case of the LHS method. However, the advantage of this approach is that the database contains individuals that may be close to the global minimum.

2.3. Metamodels for objective function prediction

The coefficient of determination R^2 was selected for assessing prediction efficiency. Although it does not guarantee that the model with the highest R^2 provides the most accurate prediction, this metric is useful for comparing different metamodels [20–23]. When combined with validation loss and training loss, it offers a good indication of whether the function is well fitted to the data. Hyperparameter tuning tools for ANN, such as Optuna [24], help to avoid overfitting by stopping the training process when the validation loss becomes greater than the training loss. This approach serves as a reliable indicator of overfitting. Prediction efficiency and the R^2 coefficient depend heavily on the input data. It is well known that data must be carefully prepared before being used in metamodeling. Several preprocessing techniques are applied to improve the quality

of input data, such as normalization, scaling, and outlier removal. Metamodels generally perform better when the input data are more uniformly distributed. In this case, the dataset showed a tendency to produce a high number of results without any damage, which led to a non-uniform distribution in the results database. Optimization based on a model trained on such data may result in high prediction errors. In this study, the primary focus is on the design space where no damage is observed. Therefore, it was essential to allocate more attention to achieve higher prediction accuracy within this specific region [25, 26]. Two types of databases were used to store the data: the first one (full) contained the full range of objective function values, while the second database (no-damage) included only design vectors for which no damage occurred (no elements were deleted during FEM simulations). The model selection logic is presented in Fig. 4.

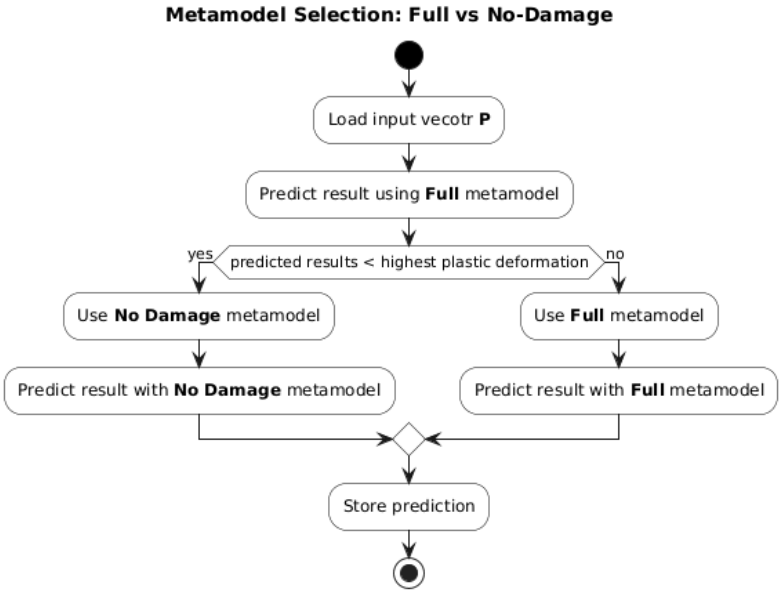


Fig. 4. Model selection logic.

When the predicted damage exceeds the threshold of the maximum possible plastic strain, the metamodel returns a prediction based on the model trained on the full database. Otherwise, if the value of the fitness function indicates that no damage occurred in the model, the prediction is made using the database that contains only undamaged individuals. This approach allows obtaining good predictions in the design space that is the most interesting from the point of view of this problem, while also obtaining a good global view of the response space using the full database.

2.4. Metamodels and hyperparameters tuning

ANN, Kriging method and PNR were used in this study. As ANNs have become more widespread in various industrial fields, new tools for hyperparameter optimization have been developed in recent years [27]. The most popular ANN hyperparameter tuning methods are grid search and random search [28]. The grid search method allows for an exhaustive search through all configurations, but it is computationally inefficient. The random search method is less accurate, as it carries the risk of missing the optimal configuration; however, it has lower computational costs. ANN has a large number of hyperparameters that must be tuned. The following activation functions were considered: rectified linear unit (ReLU), tanh, sigmoid, softmax, softplus, softsign, scaled exponential linear unit (SELU), exponential linear unit (ELU), exponential, linear, and swish. Regularizations were applied to the dataset, along with techniques such as neuron dropout. The dropout rate was treated as a tunable parameter that controls the fraction of neurons randomly 'dropped out' (set to zero) during each training iteration. Next, the number of hidden layers can be defined. Once the number of layers is set, it is important to tune the number of neurons in each hidden layer. The objective is to achieve the highest possible R^2 value while minimizing training time.

To construct the Kriging metamodel, it was essential to select the type of variogram function (variogram model). This function describes how variance changes with spatial distance between samples. Five standard models were evaluated: linear, power, Gaussian, spherical, and exponential. In cases involving a small number of variables, a grid search approach was used to explore all possible configurations [29].

When constructing a metamodel based on PNR, a key aspect of hyperparameter tuning is selecting the appropriate order (degree) of the polynomial. A polynomial degree that is too low may result in underfitting, where the model fails to capture important trends in the data [30, 31]. On the contrary, an excessively high degree increases the risk of overfitting and introduces unnecessary computational complexity. In practice, the optimal degree is typically determined using cross validation or a dedicated validation dataset. Popular hyperparameter tuning methods for PNR include grid search and random search. However, when faced with a large number of possible configurations, more advanced strategies, such as Bayesian optimization, can be advantageous, as they allow for more efficient identification of promising parameter sets. In practice, the tuning process involves iteratively testing models with different polynomial degrees and regularization parameters, and then selecting a solution that provides the best results (for example, the highest coefficient of determination R^2 with the lowest validation error). Due to the quite small database base and small number of

variables, a grid search technique was used, after checking every degree of the PNR model. To determine the most reliable PNR model, a custom ranking criterion was introduced instead of simply selecting the configuration with the highest test R^2. For each trained model (defined by a specific polynomial degree and data split), three performance metrics were computed: R^2 on the training set (R^2_{train}), validation set (R^2_{val}), and test set (R^2_{test}). The final selection was based on a composite score, defined as the mean of the three R^2 values, penalized by their pairwise absolute differences. Mathematically, the score for each model was computed as follows:

$$\text{score} = \frac{R^2_{\text{train}} + R^2_{\text{val}} + R^2_{\text{test}}}{3} - \left|R^2_{\text{train}} - R^2_{\text{val}}\right| - \left|R^2_{\text{val}} - R^2_{\text{test}}\right|. \tag{7}$$

This approach prioritizes models that not only achieve high predictive performance but also maintain consistency across all data subsets, reducing the risk of overfitting or poor generalization. The configuration with the highest score was then selected as the final model for each dataset variant (i.e., full and no-damage). This ensures that the chosen polynomial degree is both accurate and robust with respect to unseen data.

2.5. QEA optimization with metamodels

The QEA used in this study is the previously published algorithm described in [16]; the present work focuses on metamodel selection, hyperparameter tuning, and the evaluation of hybrid strategies for embedding surrogate predictions into this workflow. Running optimization based only on previously generated databases and static metamodel predictions may not represent a viable engineering solution. However, such an approach was used for testing purposes. Solving the task exclusively using QEA and evaluating the fitness function with FEM calculations leads to excessively long computational times. Previous studies have shown that a single optimization run, without metamodels, may require up to 40 hours, and repeating the process at least four times results in a total computation time of approximately 150 hours (sometimes, more repetitions are required, but the computational cost becomes too high). To reduce computational cost, a hybrid approach is proposed. In the context of the previously developed QEA workflow, a practical extension is to generate the database dynamically during the optimization process and use a metamodel to accelerate objective (fitness) function evaluation.

Two strategies are proposed, each with two phases. In phase A, the objective (fitness) function is calculated using FEM only. In phase B, the process is supported by metamodel predictions based on data collected in the first phase. A flowchart of the algorithm is shown in Fig. 5.

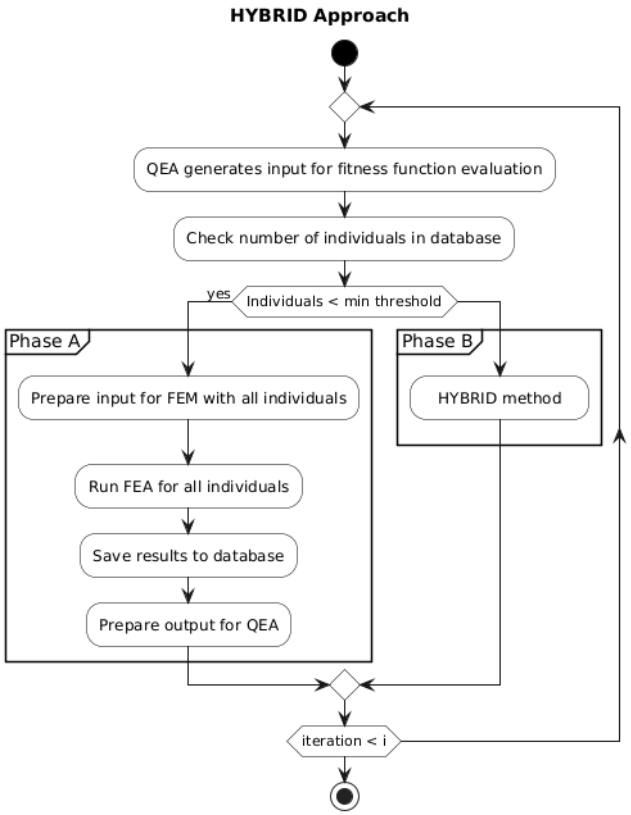


FIG. 5. Flowchart of the HYBRID algorithm.

Two different HYBRID approaches will be tested. The first approach, called HYBRID1, is based on the logic that, in each iteration of phase B, all objective (fitness) function evaluations are performed using the metamodel. The best result is then recalculated using FEM. The obtained value is used to update the database. The ANN in the next steps is trained including new solutions. The scheme in Fig. 6 shows the workflow of HYBRID1.

In contrast to HYBRID1, in the HYBRID2 approach, obligatory FEM calculations were omitted in every optimization iteration. If the best individual in a generation had already been evaluated using FEM, its result was retrieved from the database. With each new step, fewer predictions were saved until eventually none were written. Furthermore, the network was not retrained at every iteration, but only in the case where at least two new individuals were added to the database. If no new individuals were introduced, the prediction model remained unchanged. This approach allows obtaining more stable results, but in the case of new entries, the metamodel must be retrained again. This approach is presented in Fig. 7.

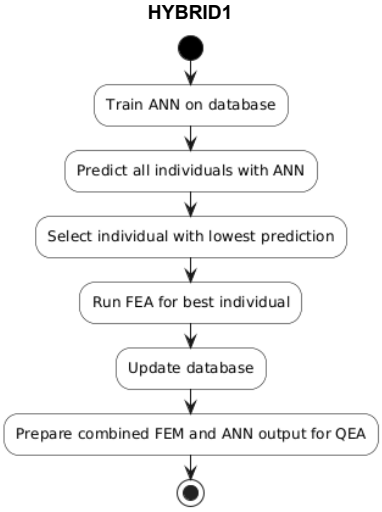


Fig. 6. Scheme of HYBRID1 approach in phase B.

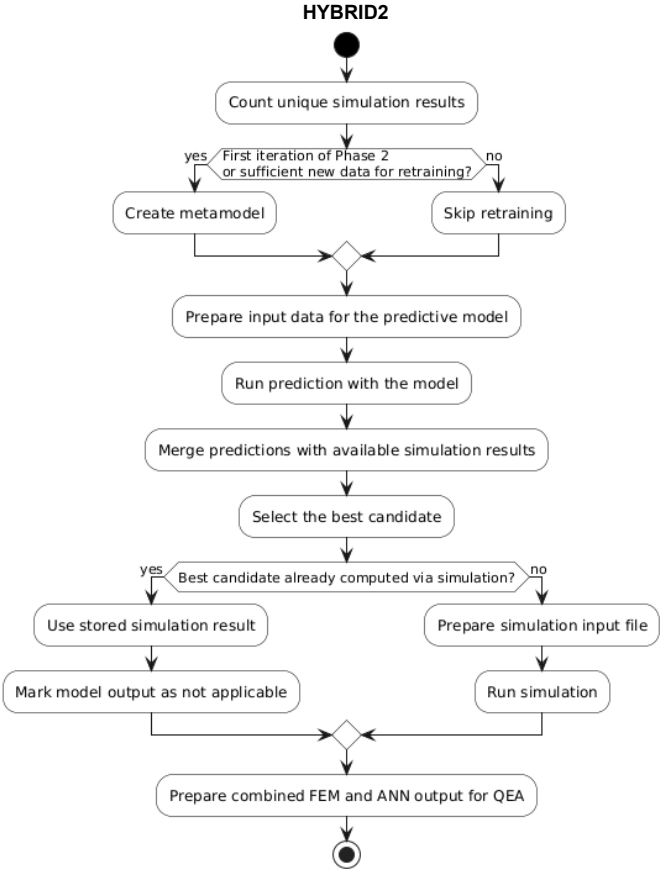


Fig. 7. Scheme of the HYBRID2 approach in phase B.

3. Numerical example description

To evaluate the approach proposed in this article, a conceptual battery design was created, defined by a design parameters vector **P** consisting of three parameters. The values of parameters are constrained in the range from 1 mm up to 3 mm. This range was selected based on practical wall-thickness limits for the considered injection-molded housing concept. In the analyzed application, the housing mass itself was not treated as the primary design driver, while manufacturability and impact resistance were treated as the main constraints. The geometry of the structure is shown in Fig. 8a. This case was intentionally selected as a representative and computationally demanding problem, allowing systematic comparison of different metamodeling techniques, datasets, and optimization strategies within a consistent framework.

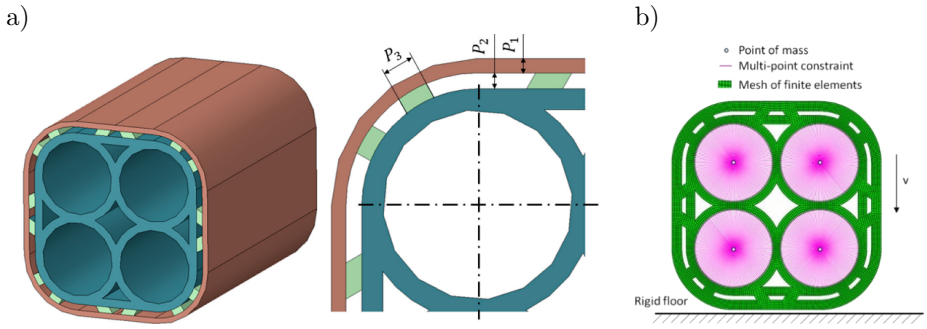


Fig. 8. a) Geometry of the battery housing, b) cell point-mass locations.

On the basis of this design, a FEM numerical model was developed. The housing was discretized with approximately 64 000 eight-node linear solid elements, using an average element size of 0.4 mm and at least four elements through the wall thickness. The battery cells were represented by 0.07 kg inertia points coupled to the housing. The inertia points were connected to the housing by a multi-node constraint, while all degrees of freedom of the analytical rigid surface were fixed. The drop event was modeled as an impact against a fully fixed analytical rigid surface, with the initial velocity prescribed normal to the surface. The enclosure material was represented by a homogenized isotropic PA6-GF30 model with Young's modulus of 9.1 GPa, Poisson's ratio of 0.34, and density of 1350 kg/m^3, combined with von Mises plasticity, bilinear hardening, and a triaxiality-dependent ductile damage initiation model. The highest plastic equivalent strain value in this material was determined as 1.014. The metamodels, described as full and no-damage, use this threshold as the switching criterion. The drop sequence included two drops in the same direction, corresponding to the two free-fall velocities: 10.3 m/s and 16.5 m/s. Further implementation details are given in [16].

The hyperparameters of the metamodels were trained on the basis of FEM results obtained for the above problem. After performing calculations for 5000 samples generated using LHS, the LHS5000 database was created. The distribution of results in this database is shown in Fig. 9, where P_1, P_2, and P_3 are design parameters and Del_El_Volume is $V_{\mathrm{del}}(\mathbf{P})$ and equivalent plastic strain (PEEQ) is $\varepsilon_{p\_\max}(\mathbf{P})$.

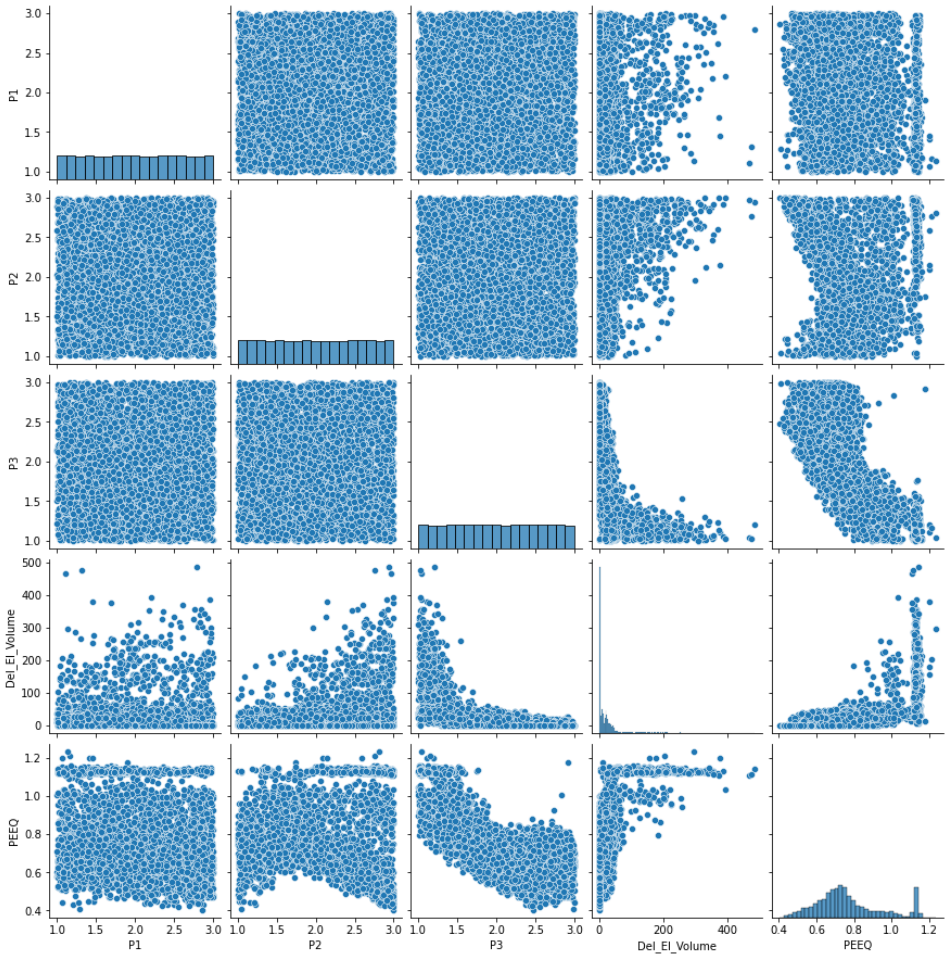


Fig. 9. Review of the LHS database.

As mentioned in the previous chapter, smaller databases were also created: LHS2500, LHS1225, LHS625, LHS312, LHS156 with 2500, 1225, 625, 312, and 156 samples, respectively.

The data obtained from multiple runs of the optimization task described in [16] were merged into a single database. The combined dataset was then randomly sorted and divided into smaller subsets. The QEA-based databases

included the following subsets: QEA1687, QEA625, QEA312, and QEA156. The largest database (QEA1687) is shown in Fig. 10.

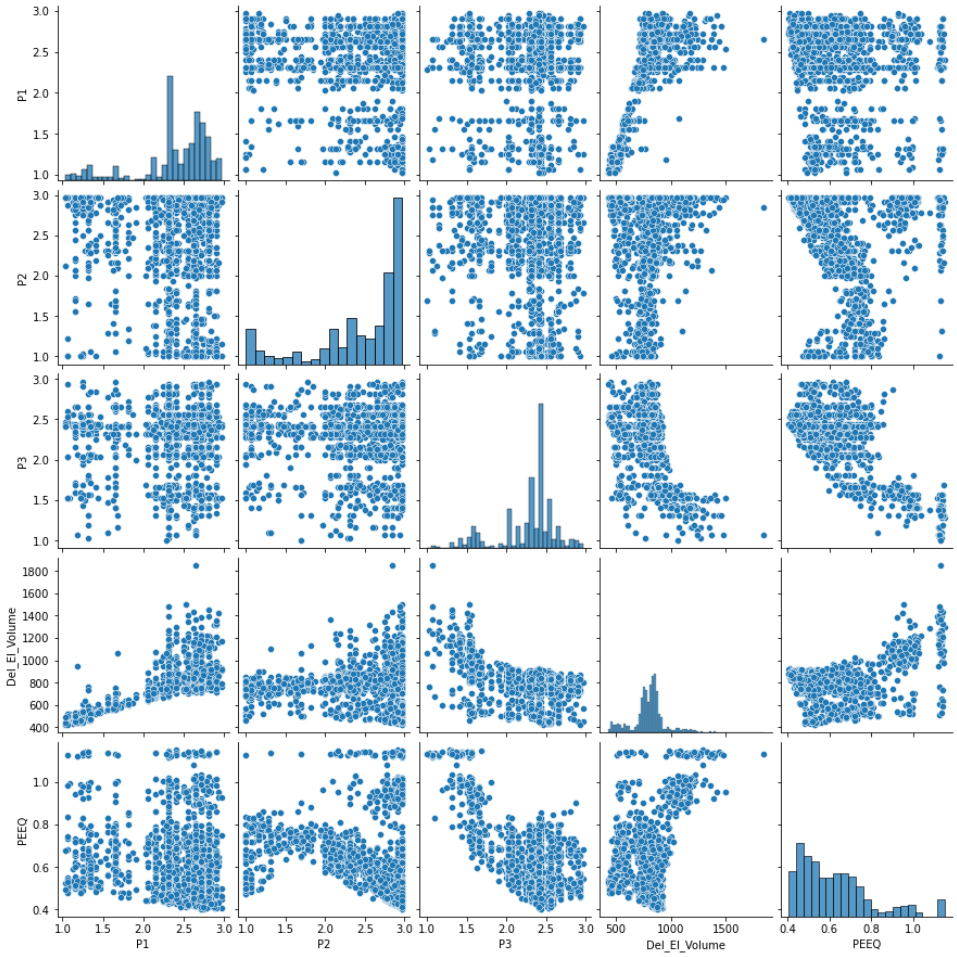


Fig. 10. Overview of the QEA1687 database.

3.1. ANN tuning of hyperparameters

To establish a universal configuration applicable to all databases, the LHS5000 database was selected due to its uniform distribution and the largest number of samples, providing a solid foundation for hyperparameter optimization. For hyperparameter tuning, the objective function FF1 – Eq. (4) was used. Regularization was applied during hyperparameter optimization (HPO), and a dropout rate of 20 % was used to randomly deactivate neurons in order to prevent overfitting. The Adam optimization algorithm [32] was used. The initial configuration consisted of a two-layer ANN with 30 neurons in each hidden layer. The

maximum number of training epochs was set to 1000, with an early stopping mechanism triggered if no improvement in validation loss was observed within 50 consecutive epochs. The data were standardized using a standard scaler and divided into three sets: training data – 80 %, validation data – 10 %, and test data – 10 %. The optimization process included 500 trials. Since Optuna adaptively allocates trials to promising regions of the search space, the number of evaluations assigned to individual activation functions was not balanced. Therefore, the results reported in Table 1 should be interpreted as a descriptive summary of the HPO process rather than as a controlled comparison with equal evaluation budgets for each activation function. The results of this study are presented in Fig. 11 and Table 1.

Table 1. Descriptive summary of activation-function results obtained during the Optuna search.

Activation function	Trials	R^2	Standard deviation R^2	Average time [s]
ReLU	347	0.8155	0.0032	64.46
Sigmoid	15	0.8135	0.0022	135.32
Swish	15	0.8108	0.0053	113.73
Softplus	15	0.8014	0.0063	119.53
ELU	16	0.7924	0.0063	82.22
Tanh	16	0.7821	0.0071	109.87
Leaky ReLU	16	0.7758	0.0085	91.34
Softmax	15	0.6212	0.0124	159.34
Linear	15	0.3628	0.0093	18.69
Hard Sigmoid	15	0.3415	0.0152	22.41
Exponential	15	0.2894	0.0185	31.57

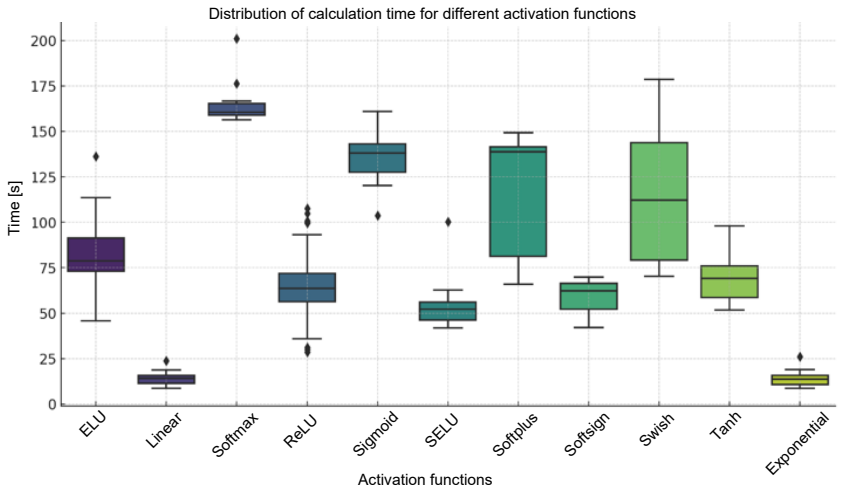


Fig. 11. Calculation time for various activation functions.

In the performed Optuna search, ReLU was associated with the highest average R^2, the lowest standard deviation, and a relatively short computation time among the frequently selected configurations. However, because the search budget was allocated adaptively and unequally across activation functions, these results should be treated as a descriptive outcome of the HPO procedure rather than as a strictly controlled comparison. For this reason, ReLU was adopted in subsequent experiments as the most promising activation function identified within the performed search. Using the activation function identified in the previous study, a new experiment was conducted to investigate the simultaneous effect of the number of layers (ranging from 1 to 4) and the number of neurons (ranging from 2 to 100). A total of 100 tests were performed. The results are presented in the following graph. The findings indicate that these hyperparameters have a minimal impact on both prediction accuracy and computational time. Networks with a single hidden layer required significantly more computation time. Networks with three or four layers performed faster and delivered better results, but were less stable and occasionally produced low-quality predictions, as illustrated in Fig. 12.

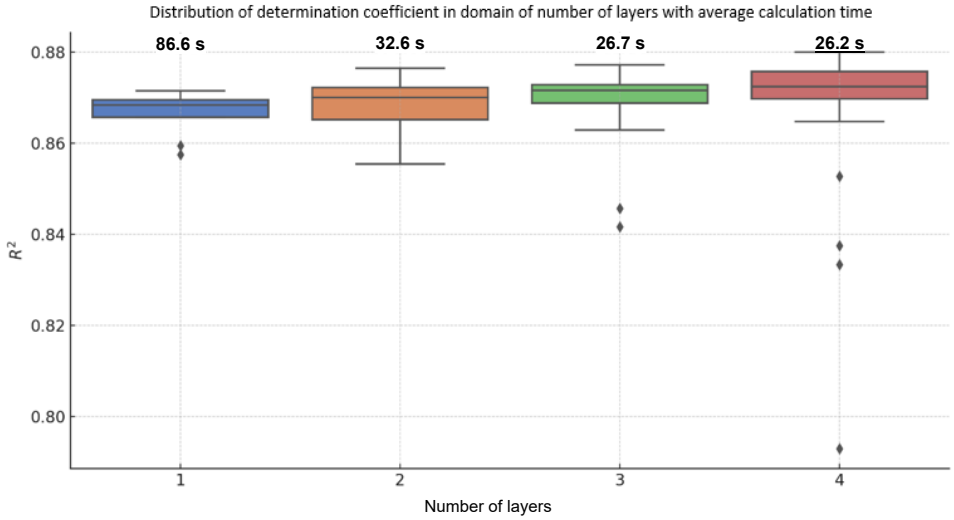


FIG. 12. Results for different numbers of hidden layers and computation time.

Since there was no significant difference in average training time and the model remained stable, subsequent tests used networks with two hidden layers. To determine the optimal number of neurons in each layer, the grid search method was applied. This method evaluates all possible combinations and enables visualization of R^2 values using a heat map. The results are shown in Fig. 13.

To evaluate the impact of different validation sets on training time and model prediction accuracy, an experiment was conducted using five distinct initializa-

tions of the validation, test, and training datasets. These datasets remained consistent within each individual test but varied between repetitions within the same test configuration. In total, 1800 neural network training sessions were performed. The results are illustrated in Fig. 13 and Fig. 14.

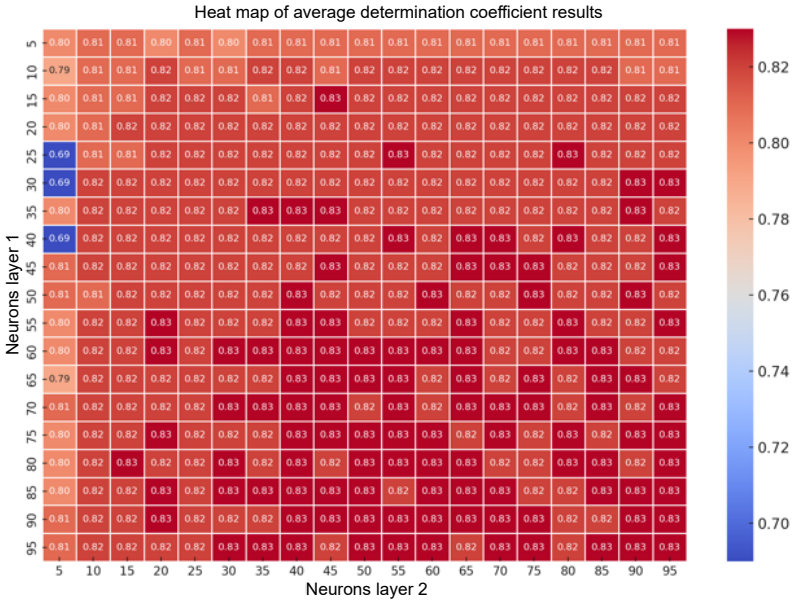


Fig. 13. Heat map of the coefficient of determination for various numbers of neurons in layers.

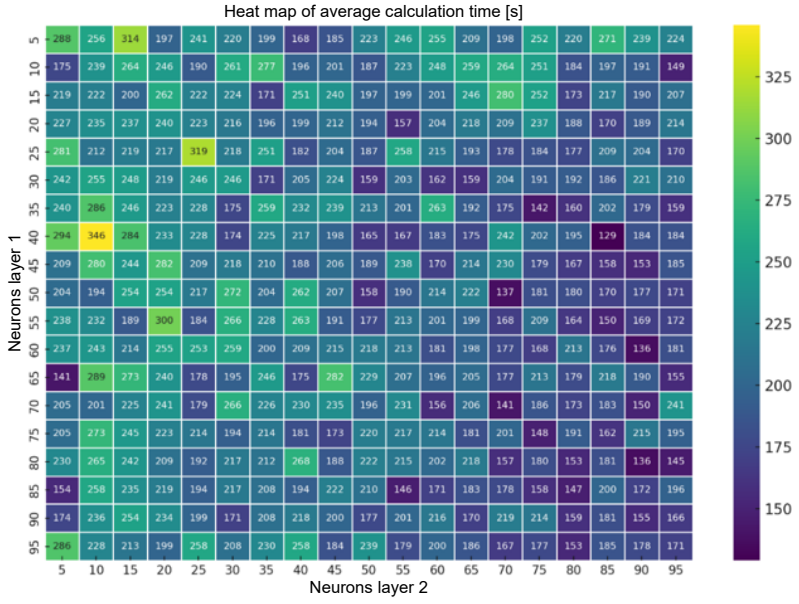


Fig. 14. Heat map of computation time for various numbers of neurons in layers.

For subsequent tests, it was decided that a configuration with at least 50 neurons in each layer would ensure stability and maintain a short computation time. An example of learning curves for FF1 is shown in Fig. 15.

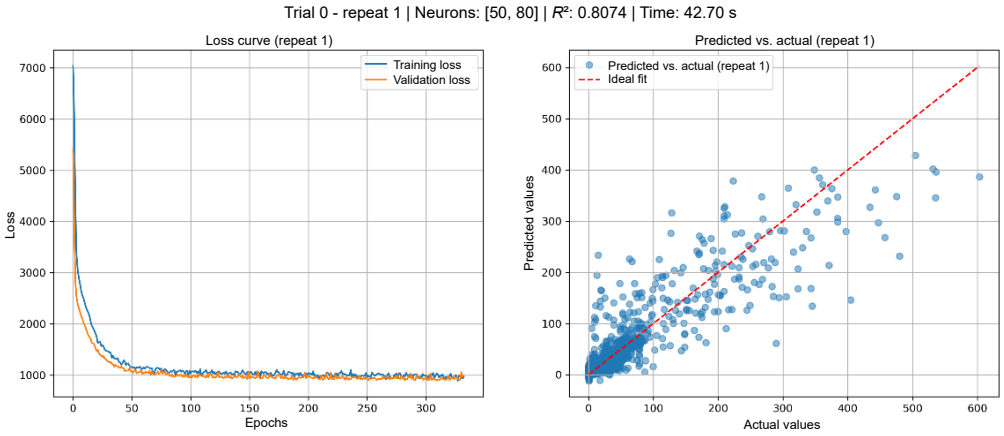


Fig. 15. Learning curves for FF1.

The parameters of objective function FF2 – Eq. (5) were adjusted by setting $a = 0.3194$ and $b = 1.014$, so that the function equals 1.014 at zero volume, which corresponds to the maximum plastic deformation $\varepsilon_{p\_\max}(\mathbf{P})$ value which can be obtained for this kind of material. At the maximum volume of approximately 500 mm^3, the function approaches a value of about 3 (Fig. 16).

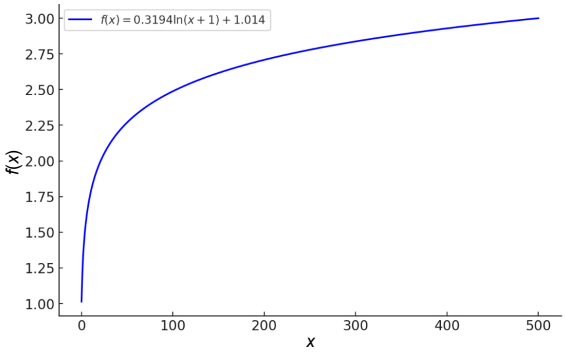


Fig. 16. Selection of parameters for FF2 to obtain a compact distribution of results.

The graph of the predicted vs. actual values shows that the values have a smaller spread, which is visible in Fig. 17.

Both learning curves show that there exists a problematic region between damage and no-damage predictions. Therefore, it was decided to use a separate ANN for prediction from the full model and the no-damage database.

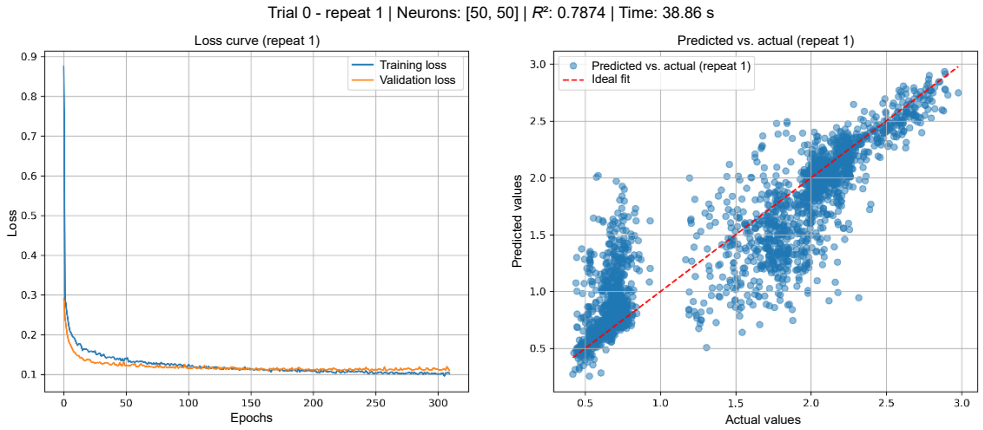
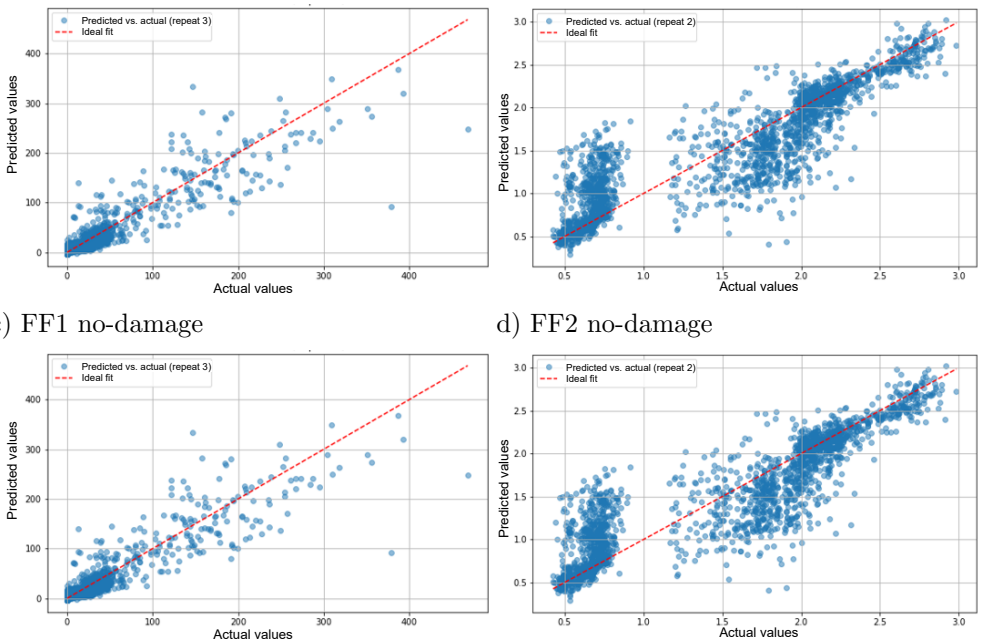


Fig. 17. Learning curves of FF2.

Results of training with separate ANN models on both full and no-damage datasets, with comparison for FF1 and FF2, are shown in Fig. 18.

a) FF1 full

b) FF2 full



c) FF1 no-damage

d) FF2 no-damage

Fig. 18. Example of learning curves for full (a and b) and no-damage models (c and d).

The next step was to determine whether different datasets, such as QEA and smaller LHS subsets, would yield similar and good results. The following table presents a set of results obtained from training the ANN five times for each dataset. The average values were recorded in Table 2.

TABLE 2. R^2 for different fitness functions and sizes of databases.

Database	R^2		Val loss		Train loss	
	Full	No-damage	Full	No-damage	Full	No-damage
ANN with FF1 based on the LHS databases						
LHS5000	0.8311542	0.7491974	426.7850708	0.0022338	452.4532778	0.0021394
LHS2500	0.824483	0.765545	445.712219	0.002176	377.840295	0.002113
LHS1250	0.837276	0.740888	455.713214	0.002262	527.78225	0.002236
LHS625	0.757262	0.568795	813.327331	0.003634	1092.683227	0.004399
LHS312	0.826518	0.625481	583.259711	0.003101	660.375433	0.005706
LHS156	0.635194	−0.607225	731.145435	0.011335	1519.446338	0.013695
ANN with FF2 based on the LHS databases						
LHS5000	0.779647	0.756274	0.113400	0.002171	0.109448	0.002031
LHS2500	0.753563	0.763015	0.127057	0.002203	0.128848	0.002258
LHS1250	0.733327	0.729156	0.135273	0.002366	0.140219	0.002363
LHS625	0.705947	0.587907	0.157150	0.003478	0.170239	0.004043
LHS312	0.733171	0.328765	0.141381	0.005361	0.158526	0.009317
LHS156	0.641887	−1.0251	0.173655	0.013695	0.141143	0.016411
ANN with FF1 based on the QEA databases						
QEA1687	0.748013	0.832073	197.507169	0.001692	255.213092	0.002015
QEA625	0.517468	0.847087	285.901752	0.001646	139.069743	0.001530
QEA312	0.519403	0.866713	126.929550	0.001416	166.399509	0.001367
QEA156	0.327259	0.460945	70.565874	0.006220	96.116487	0.006020
ANN with FF2 based on the QEA databases						
QEA1687	0.750640	0.860533	0.115640	0.001528	0.117122	0.001543
QEA625	0.756393	0.830961	0.115566	0.001924	0.133548	0.002240
QEA312	0.663904	0.776071	0.163220	0.002354	0.181552	0.002585
QEA156	0.629303	0.245422	0.206355	0.008436	0.131287	0.007253

The differences between the fitness functions appear to be relatively small; however, FF2 demonstrated higher R^2 values for smaller datasets, which is particularly noteworthy in this context. Therefore, subsequent tests were conducted using FF2.

After the hyperparameter tuning process was completed, the next step was to apply this knowledge to assess prediction performance. The study involved two primary computational tasks:

- optimization using the full input set, which consisted of 5000 individuals generated via LHS, and
- evaluation of the prediction model performance using only data obtained from the QEA process, which consisted of 1687 individuals.

Table 3 presents results from 16 separate optimization tests. These tests were carried out for four different prediction models. Each model was checked four times using the QEA algorithm. The aim of the analysis was to examine the variability of results across repeated runs of the same model.

Table 3. ANN prediction vs. FEM results.

Model [R^2]	Trial	Result	Individual	FEM result
QEA FF2 based on LHS5000				
Full = 0.792 ND = 0.78	1	0.439186	2.375, 2.9375, 2.4375	0.44241
	2	0.435265	2.8125, 2.9375, 2.625	0.44878
	3			
	4	0.439186	2.375, 2.9375, 2.4375	0.44241
Full = 0.7847 ND = 0.771	1	0.461702	2.9375, 2.9375, 2.5	0.46484
	2			
	3			
	4			
Full = 0.7844 ND = 0.754	1	0.45215	2.25, 2.9375, 2.5	0.445811
	2			
	3			
	4			
Full = 0.7820 ND = 0.751	1	0.431665	2.9375, 2.9375, 2.5	0.46484
	2			
	3			
	4			
QEA FF2 based on QEA1687				
Full = 0.7213 ND = 0.8713	1	0.437105	2.75, 2.9375, 2.4375	0.41423
	2			
	3			
	4			
Full = 0.7140 ND = 0.8761	1	0.439528	2.6875, 2.9375, 2.4375	0.411295
	2			
	3	0.439374	2.5625, 2.9375, 2.5	0.42695
	4	0.438613	2.625, 2.9375, 2.4375	0.42717
Full = 0.7260 ND = 0.8721	1	0.438376	2.75, 2.9375, 2.5	0.434021
	2			
	3			
	4			
Full = 0.7396 ND = 0.9749	1	0.448706	2.6875, 2.9375, 2.5	0.42717
	2			
	3			
	4			

Figure 19 shows the fitness function value as a function of iteration number.

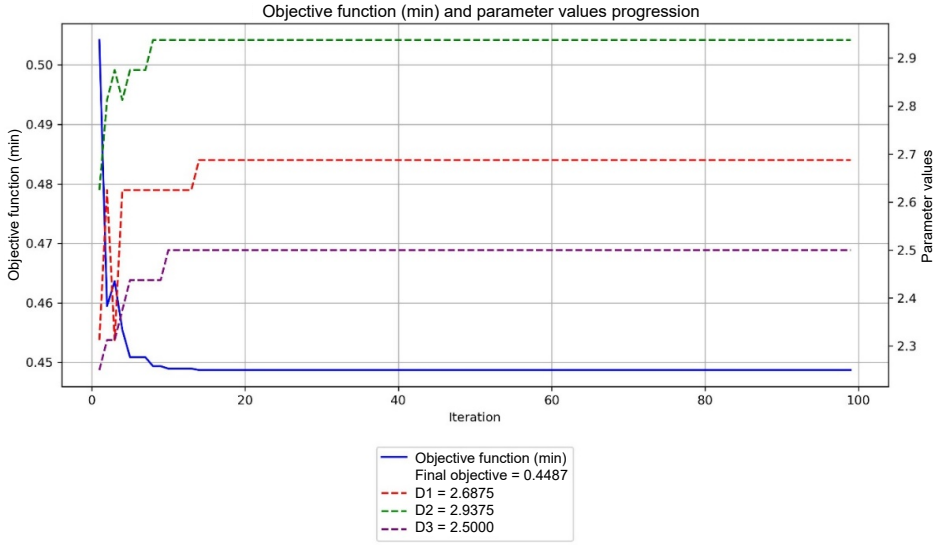


FIG. 19. Fitness function value as a function of iteration number.

3.2. KRIGING-BASED METAMODEL

Each variogram model – linear, power, Gaussian, spherical, and exponential – was evaluated five times using different training and validation dataset splits. Hyperparameter tuning was performed using the Optuna library in GridSampler mode, which produced the average prediction quality (R^2) for each configuration. The variogram model that achieved the best fit (linear) was selected for the final metamodel, as shown in Fig. 20.

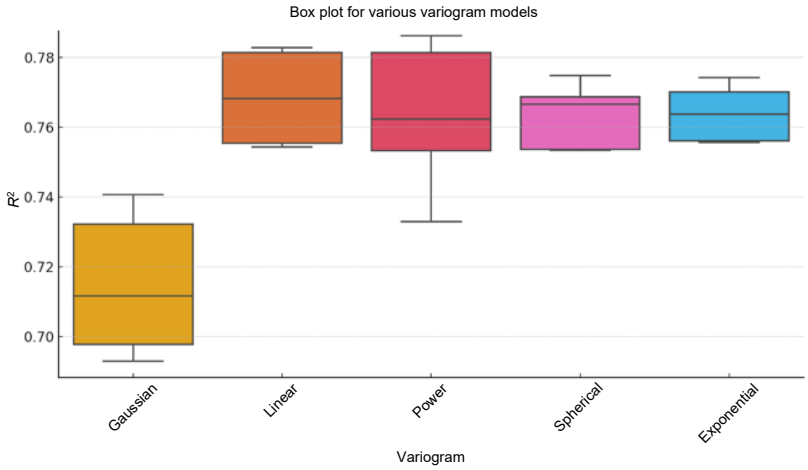


FIG. 20. HPO results for Kriging with different variogram models.

Table 4 shows the results obtained using Kriging on both the LHS and QEA databases of different sizes.

Table 4. R^2 values for several sizes of the LHS and QEA databases.

Database	R^2		Val loss		Train loss	
	Full	No-damage	Full	No-damage	Full	No-damage
Kriging with FF2 based on the LHS databases						
LHS5000	0.782909	0.756464	0.214571	0.030546	0.334160	0.046494
LHS2500	0.754501	0.761495	0.234398	0.030487	0.355564	0.046969
LHS1250	0.760380	0.773227	0.241413	0.030536	0.348399	0.044433
LHS625	0.766255	0.538562	0.248900	0.045064	0.353607	0.061690
LHS312	0.734848	0.618931	0.258842	0.041498	0.374150	0.055653
LHS156	0.660231	0.228298	0.307563	0.057847	0.405318	0.072067
Kriging with FF2 based on the QEA databases						
QEA1687	0.773371	0.858205	0.202413	0.025761	0.323797	0.039368
QEA625	0.768539	0.816137	0.213611	0.033606	0.330643	0.045771
QEA312	0.737385	0.772807	0.242555	0.033486	0.357599	0.048491
QEA156	0.674119	0.656937	0.298575	0.041870	0.424546	0.061597

The QEA tasks relied exclusively on predictions from the Kriging metamodel. Four different models were tested, each trained and evaluated on a dis-

Table 5. Kriging predictions vs. FEM.

Model [R^2]	Result	Individual	FEM result
QEA FF2 based on LHS5000			
Full = 0.7954 ND = 0.8034	0.43023	2.6875, 2.9375, 2.5	0.42717
Full = 0.7795 ND = 0.7269	0.428881	2.8125, 2.9375, 2.4375	0.42132
Full = 0.7886 ND = 0.7942	0.423571	2.625, 2.9375, 2.5	0.42642
Full = 0.7885 ND = 0.78	0.43848	2.8125, 2.9375, 2.4375	0.42132
QEA FF2 based on QEA1687			
Full = 0.7993 ND = 0.8657	0.427507	2.6875, 2.9375, 2.5	0.42717
Full = 0.8023 ND = 0.870	0.425319	2.6875, 2.9375, 2.5	0.42717
Full = 0.7857 ND = 0.8735	0.422793	2.75, 2.9375, 2.5	0.43402
Full = 0.7957 ND = 0.8643	0.420768	2.75, 2.9375, 2.5	0.43402

tinct dataset. The same model was not tested multiple times because previous experiments indicated that the results were highly consistent. The results are presented in Table 5.

On average, the Kriging method produced better results. However, the ANN surrogate models identified individuals with lower values of objective functions. In general, the performance of both methods was similar.

3.3. PNR-based metamodel

Hyperparameter tuning of the PNR model showed a tendency that, for the largest LHS5000 database, increasing the polynomial degree in a lower range resulted in a model that was not well fitted to the data. Later, the model achieved a maximum value of R^2 on both the training and validation sets. For higher polynomial degrees, the model seemed to be overfitted, and R^2 was often below 0. Figure 21 shows the distribution of the coefficient of determination R^2 of the training set and the validation set. Models with R^2 below zero were omitted from the visualization.

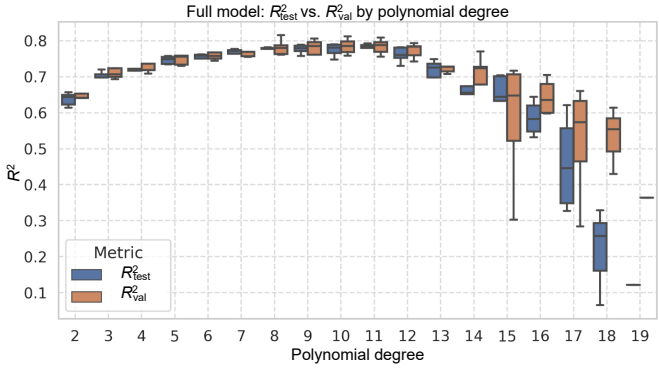


Fig. 21. PNR degree selection for the LHS5000 database in the full model.

The same situation happened for the no-damage model, as shown in Fig. 22.

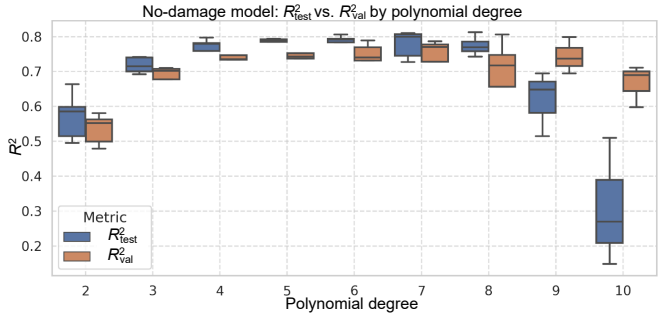


Fig. 22. PNR degree selection for the LHS5000 database in no-damage model.

Table 6 shows the results obtained for the PNR models using different databases. The observed behavior seems to be quite similar to the previously investigated metamodeling techniques.

Table 6. PNR results for different databases.

Database	R^2 training		R^2 validation		R^2 test	
	Full	No-damage	Full	No-damage	Full	No-damage
PNR with FF2 based on the LHS databases						
LHS5000	0.796782	0.790415	0.787567	0.791222	0.779578	0.789302
LHS2500	0.806827	0.802475	0.762755	0.831658	0.772246	0.831917
LHS1250	0.772573	0.75779	0.775060	0.761361	0.760191	0.775766
LHS625	0.735603	0.730839	0.723093	0.732130	0.725574	0.747068
LHS312	0.749152	0.629371	0.739849	0.581701	0.758322	0.532763
LHS156	0.760563	0.791089	0.776078	0.588699	0.792646	0.496165
Database	R^2		Val loss		Train loss	
	Full	No-damage	Full	No-damage	Full	No-damage
PNR with FF2 based on the QEA databases						
QEA1687	0.778242	0.862380	0.799098	0.869001	0.742744	0.873709
QEA625	0.751164	0.845151	0.753742	0.809160	0.757736	0.829696
QEA312	0.769839	0.879970	0.777884	0.868079	0.731634	0.861894
QEA156	0.661729	0.906574	0.550962	0.881134	0.613979	0.931966

It is worth noticing that the polynomial degree tends to be lower for smaller databases. The creation of PNR models does not take long computational time; so, it was possible to use a cropped grid search technique to obtain the model with the best score. The selected degrees for all databases are shown in Table 7.

Table 7. PNR degrees for the best models.

Database	PNR		Database	PNR	
	Full	No-damage		Full	No-damage
LHS5000	8	5	QEA1687	6	4
LHS2500	9	6	QEA625	3	3
LHS1250	6	3	QEA312	4	3
LHS 625	3	3	QEA156	2	3
LHS 312	3	2			
LHS 156	3	2			

After establishing a method for the selection of the PNR degree, the QEA optimization process was submitted to check the results of optimization based only on predictions from the PNR metamodel. The optimization results are shown in Table 8.

Table 8. PNR predictions vs. FEM.

Model [R^2]	Result	Individual	FEM result
PNR FF2 based on LHS5000			
Full = 0.77 ND = 0.789	0.4466	2.5625, 2.9375, 2.625	0.4858
Full = 0.79 ND = 0.8	0.4377	2.6875, 2.9375, 2.5	0.427044
Full = 0.79 ND = 0.8	0.4282	2.6875, 2.9375, 2.5625	0.43345
Full = 0.78 ND = 0.81	0.4456	2.625, 2.9375, 2.5625	0.4306
PNR FF2 based on QEA1687			
Full = 0.77 ND = 0.84	−0.36	1, 1, 2.9375	0.41940
Full = 0.76 ND = 0.88	0.4233	2.6875, 2.9375, 2.5625	0.43345
Full = 0.78 ND = 0.84	0.4245	2.6875, 2.9375, 2.5625	0.43345
Full = 0.73 ND = 0.87	0.4191	2.625, 2.9375, 2.5625	0.4306
Full = 0.76 ND = 0.86	−11.375	2.9375, 2.9375,1	325.659

Although several PNR predictions were close to the FEM results, the method showed clear instability, including failed optimization attempts and non-physical extrapolation errors. For this reason, it is recommended to implement an additional mechanism that performs multiple runs and selects only reliable results. Alternatively, it is recommended to replace PNR with different metamodeling techniques. It would be a good direction to develop this approach in future work.

3.4. Results of optimization with the selected metamodel

The main purpose of this paper was to present the results of QEA optimization supported by metamodels. Considering the results obtained within the present evaluation framework, ANN was selected as the most practical metamodel for the subsequent hybrid tests. Although Kriging and PNR also produced competitive results in selected cases, the comparison was not based on a fully uniform repetition protocol; therefore, these observations should be treated as indicative rather than statistically conclusive. The HYBRID1 and HYBRID2 approaches were tested. The minimum threshold for starting the HYBRID approach was set to 300 individuals, based on earlier evaluations of prediction efficiency for smaller datasets. The results obtained for the HYBRID1 approach are shown in Table 9. Four optimization runs were performed. Only the first run started with an empty database, whereas the subsequent runs were warm-

Table 9. Results of the HYBRID1 approach.

Attempt	Result	Individual	Initial data quantity	Calculation time
HYBRID 1 full task FF2				
1	0.404056	2.84375, 2.96875, 2.53125	0	29 h
2	0.404975	2.75, 2.96875, 2.4375	345	3 h 30 min
3	0.404056	2.84375, 2.96875, 2.53125	360	3 h 40 min
4	0.404056	2.84375, 2.96875, 2.53125	377	3 h 15 min

started from progressively enlarged databases containing previously computed FEM evaluations.

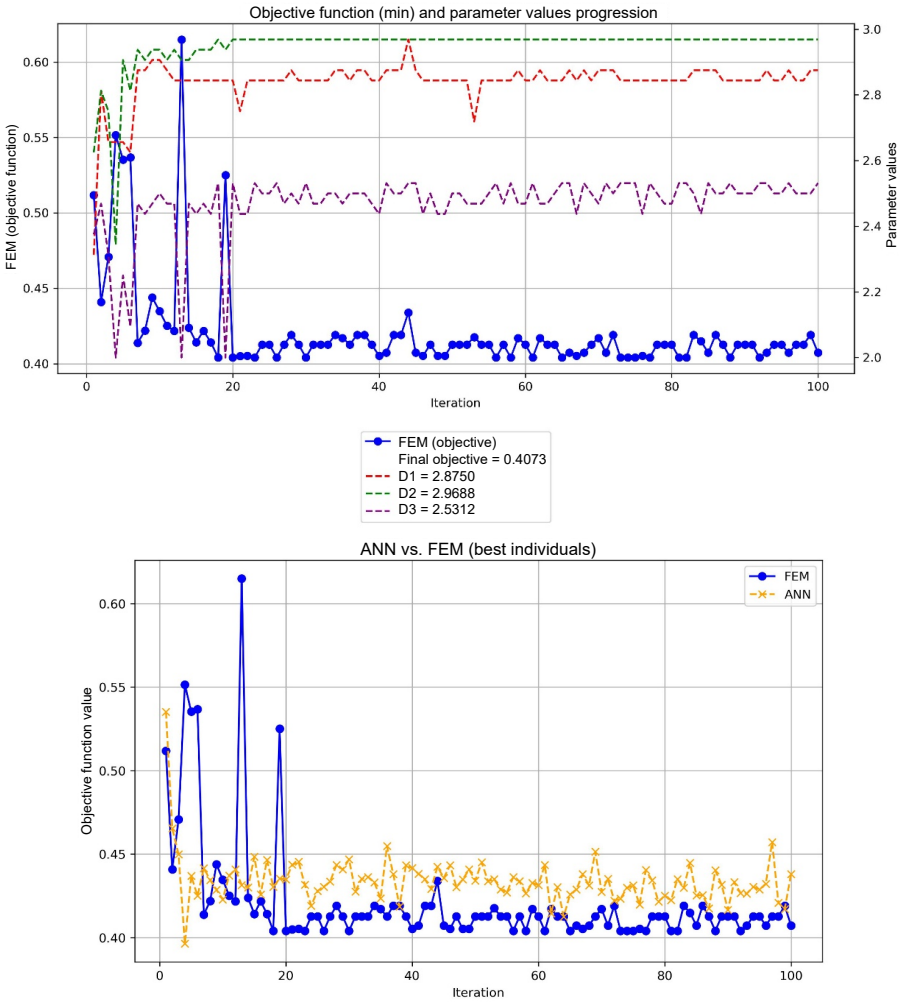


Fig. 23. Example optimization history from a HYBRID1 run.

This result appears to be quite efficient; however, some interesting phenomena occurred throughout the entire optimization attempt, as illustrated in Fig. 23. In particular, fluctuations in the results are clearly visible.

This behavior is likely caused by the fact that the same minimum fitness function value may correspond to different function outputs for the same input vector. Additionally, the optimization algorithm explores regions of the design space based on directions suggested by the metamodel. When a result is identified as the global minimum, it indicates that its fitness function value is the lowest observed among all evaluated solutions (Fig. 24).

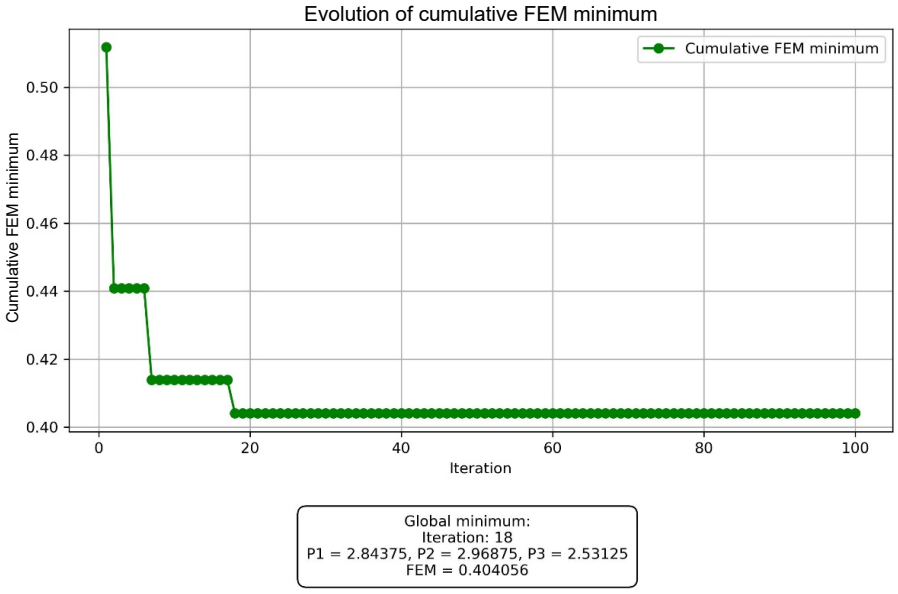


FIG. 24. Global minimum of the fourth HYBRID1 run.

The purpose of evaluating the HYBRID2 approach was to examine whether omitting certain unnecessary calculations could lead to faster results and improve the consistency and stability of the results. The results of these runs are presented in Table 10.

TABLE 10. Results of the HYBRID2 run.

Attempt	Result	Individual	Initial data quantity	Calculation time
HYBRID2 FF2 full task				
1	0.404975	2.75, 2.96875, 2.4375	0	25 h
2	0.442153	2.625, 2.84375, 2.34375	307	1 h 5 min
3	0.404975	2.75, 2.96875, 2.4375	313	40 min
4	0.404975	2.75, 2.96875, 2.4375	316	1 h 20 min

In the reported runs, HYBRID2 showed a steadier optimization course (Fig. 25). However, due to the limited number of runs and the warm-started character of runs 2–4, this observation should be treated as qualitative rather than statistically conclusive. The best-performing individual identified in this approach was slightly inferior to that obtained using the HYBRID1 method. It should also be noted that the best individuals repeatedly approached the upper bound of parameter P_2. In practical terms, this indicates that, within the present damage-oriented formulation and the assumed manufacturable

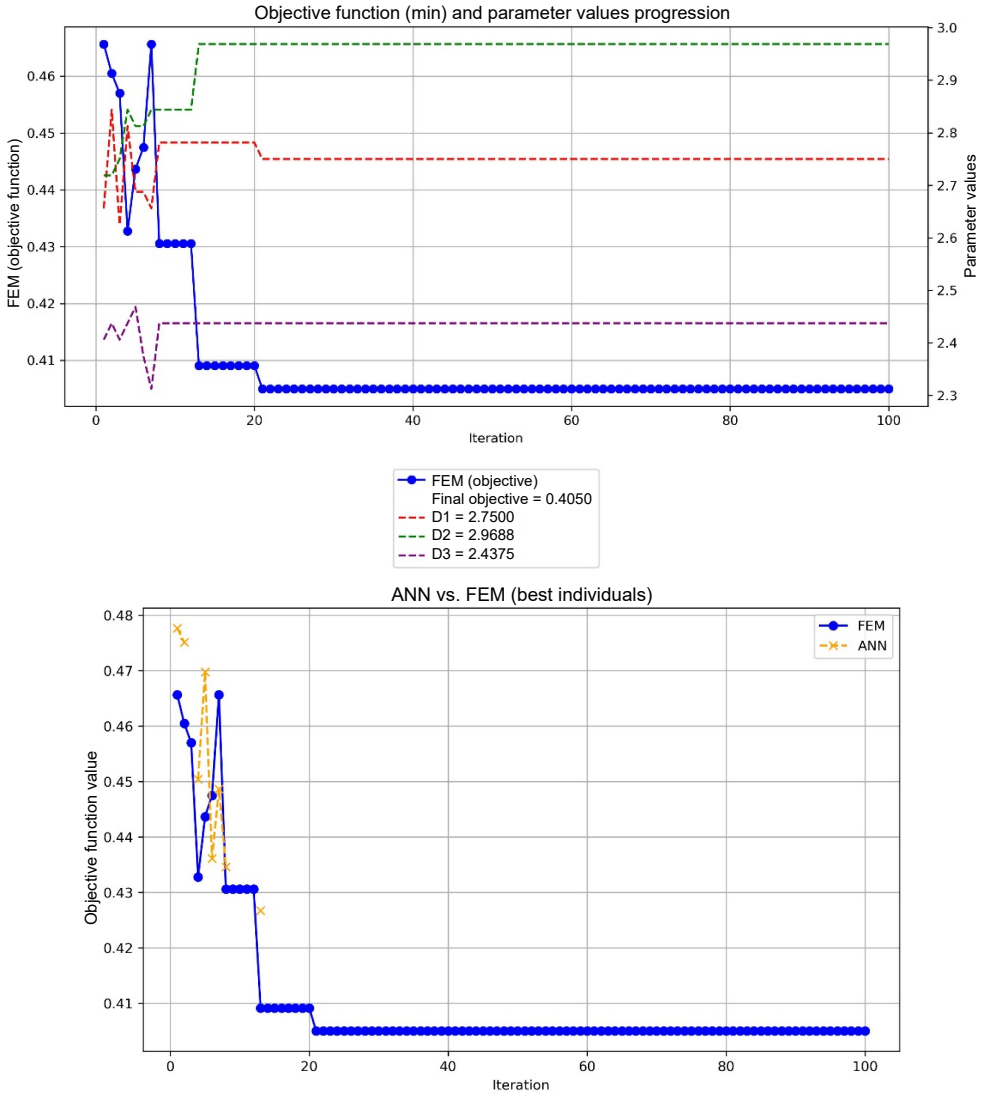


Fig. 25. Example optimization process of HYBRID2.

thickness range, increasing this wall-thickness parameter remained beneficial for impact resistance. Therefore, the obtained optimum should be interpreted as a boundary-constrained result for the analyzed housing concept rather than as a general mass-efficiency trade-off.

4. Conclusions

The presented hybridization of QEA with metamodels significantly reduces computational cost in the task of multiple crash simulations of battery enclosures while maintaining the quality of design decisions. Compared to the baseline QEA+FEM (approximately 40 hours per single run and ∼150 hours for a full series), integrating metamodels shortens the computation time considerably. Among the tested metamodels, ANN showed the most promising overall performance within the present study. Kriging provided comparable average prediction quality in several cases, whereas PNR showed instability in selected runs. Because the evaluation protocol was not fully uniform across all metamodels, these observations should be treated as indicative rather than as a definitive statistical ranking.

The objective function design played a crucial role: variant FF2, through the logarithmic transformation of the volume-related term for removed elements, reduced the range of values and alleviated the issue of discreteness near the 'damage/no-damage' threshold. This resulted in higher R^2 for smaller datasets and justified the use of FF2 under an iterative data enrichment. The use of two ANN models (full and no-damage), with switching based on a damage threshold, improved fitting in the relevant design region and stabilized learning – particularly beneficial in the transition zone.

The comparison of hybrid strategies indicates a trade-off between objective value and runtime. In the reported runs, HYBRID1 yielded the lowest objective values, whereas HYBRID2 shortened runtime and showed a steadier optimization. Due to the limited number of runs and the warm-started setup, these differences should be interpreted as indicative for the analyzed workflow rather than as statistically conclusive evidence of the superiority of one stochastic variant over the other. HYBRID1, which enforced FEM verification of the best individual in each phase B iteration and frequent retraining of the network, delivered the best outcomes (objective values ∼0.404–0.405 across four runs), although with some performance fluctuations. Its runtime dropped from ∼29 hours for an initial empty run to ∼3–4 hours for restarts using a pre-collected base of 345–377 individuals. HYBRID2, with fewer mandatory FEM recalculations and less frequent retraining, offered a more stable performance and reduced runtime to ∼40–80 minutes when resumed from a database of 307–316 individuals, albeit with a slightly weaker best individual (e.g., 0.404975 vs. 0.404056). The mini-

mum threshold for activating the hybrid mode was pragmatically set at $\geq$300 individuals, based on the observed prediction reliability for smaller datasets. Since the optimizer-level comparison in this study is limited to QEA+FEM and QEA-based hybrid variants, the conclusions presented here are restricted to the QEA framework.

Within the tested QEA-based framework and for the analyzed battery-housing problem, the combination of QEA, ANN with two networks (full and no-damage), and the FF2 objective function proved to be the most useful of the tested variants. These conclusions refer to the present workflow and test case, and should not be interpreted as a general ranking of surrogate-assisted stochastic optimization methods. Further improvements may involve incorporating uncertainty estimation and active sampling, formalizing the switching criteria between the full and no-damage models, extending validation with rank-based metrics and absolute errors, and either enhancing the robustness of PNR or replacing it with an ANN/Kriging combination in production scenarios. In summary, for the studied optimization problem, hybridizing the existing QEA workflow with ANN significantly reduces FEM computational cost while maintaining decision quality and repeatability at a practically useful level.

It should be emphasized that the conclusions drawn in this study are based on a single, representative engineering problem. While the obtained results demonstrate the effectiveness of the proposed hybrid optimization framework, further validation using different types of structures, loading conditions, and design spaces is required to fully assess its general applicability.

References

1. Sebastjan P. , Kuś W., Method for parameter tuning of hybrid optimization algorithms for problems with high computational costs of objective function evaluations, *Applied Sciences*, **13**(10): 6307, 2023, https://doi.org/10.3390/app13106307.

2. Sebastjan P. , Kuś W., For forged automotive components using hybrid optimization techniques, *Computer Methods in Materials Science*, **21**(2): 63–74, 2021, https://doi.org/10.7494/cmms.2021.2.0746.

3. Burczyński T., Kuś W., Beluch W., Długosz A., Poteralski A., Szczepanik M., *Intelligent Computing in Optimal Design*, Springer International Publishing, Cham, 2020, https://doi.org/10.1007/978-3-030-34161-9.

4. Jarosz J., Długosz A., Shape optimization of the muffler shield with regard to strength properties, *Engineering Transactions*, **71**(3): 351–366, 2023, https://doi.org/10.24423/engtrans.3093.20230727.

5. Park H.-S., Dang X.-P., Structural optimization based on CAD–CAE integration and metamodeling techniques, *Computer-Aided Design*, **42**(10): 889–902, 2010, https://doi.org/10.1016/j.cad.2010.06.003.

6. Fatiha I.C., Santosa S.P., Widagdo D., Pratomo A.N., Design optimisation of metastructure configuration for lithium-ion battery protection using machine learning methodology, *Batteries*, **10**(2): 52, 2024, https://doi.org/10.3390/batteries10020052.
7. Rajput S.P.S., Datta S., A review on optimization techniques used in civil engineering material and structure design, *Materials Today: Proceedings*, **26**: 1482–1491, 2020, https://doi.org/10.1016/j.matpr.2020.02.305.
8. Kuś W., Mucha W., Memetic inverse problem solution in cyber-physical systems, [in:] *Advances in Technical Diagnostics*, Springer International Publishing, Cham, 2018, pp. 335–341, https://doi.org/10.1007/978-3-319-62042-8_30.
9. Keane A.J., Scanlan J.P., Design search and optimization in aerospace engineering, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **365**(1859): 2501–2529, 2007, https://doi.org/10.1098/rsta.2007.2019.
10. AI and Machine Learning in Simulation, Dassault Systèmes, 2024, https://www.3ds.com/products/simulia/ai-and-machine-learning-simulation.
11. Ansys AI – AI-Augmented Simulation Technology, https://www.ansys.com/ai.
12. AI and Machine Learning for Engineering Design Generation and Optimization, Default, https://altair.com/resource/ai-and-machine-learning-for-engineering-design-generation-and-optimization.
13. Ma C., Xu C., Souri M., Hosseinzadeh E., Jabbari M., Multi-objective optimisation of the battery box in a racing car, *Technologies*, **12**(7): 93, 2024, https://doi.org/10.3390/technologies12070093.
14. Mucha W., Application of dynamic condensation for model order reduction in real-time hybrid simulations, *Meccanica*, **58**(7): 1409–1425, 2023, https://doi.org/10.1007/s11012-023-01675-0.
15. Jin R., Chen W., Simpson T.W., Comparative studies of metamodelling techniques under multiple modelling criteria, *Structural and Multidisciplinary Optimization*, **23**(1): 1–13, 2001, https://doi.org/10.1007/s00158-001-0160-4.
16. Rurański A., Kuś W., The quantum-inspired evolutionary algorithm in the parametric optimization of lithium-ion battery housing in the multiple-drop test, *Batteries*, **10**(9): 308, 2024, https://doi.org/10.3390/batteries10090308.
17. Bai Y., Wierzbicki T., A new model of metal plasticity and fracture with pressure and Lode dependence, *International Journal of Plasticity*, **24**(6): 1071–1096, 2008, https://doi.org/10.1016/j.ijplas.2007.09.004.
18. Dean A., Morris M., Stufken J., Bingham D., *Latin Hypercubes and Space-Filling Designs*, Chapman and Hall/CRC, 2015, pp. 613–646, https://doi.org/10.1201/b18619-27.
19. Zhu H., Liu L., Long T., Peng L., A novel algorithm of maximin Latin hypercube design using successive local enumeration, *Engineering Optimization*, **44**(5): 551–564, 2012, https://doi.org/10.1080/0305215X.2011.591790.
20. Zhong H. *et al.*, Metamodeling for policy simulations with multivariate outcomes, *Medical Decision Making*, **42**(7): 872–884, 2022, https://doi.org/10.1177/0272989X221105079.
21. Iooss B., Boussouf L., Feuillard V., Marrel A., Numerical studies of the metamodel fitting and validation processes, *International Journal on Advances in Systems and Measurements*, **3**(1 & 2): 11–21, 2010.

22. Kalita K., Chakraborty S., Madhu S., Ramachandran M., Gao X.-Z., Performance analysis of radial basis function metamodels for predictive modelling of laminated composites, *Materials*, **14**(12): 3306, 2021, https://doi.org/10.3390/ma14123306.

23. Williams B., Cremaschi S., Novel tool for selecting surrogate modeling techniques for surface approximation, *Computer Aided Chemical Engineering*, **50**: 451–456, 2021, https://doi.org/10.1016/B978-0-323-88506-5.50071-1.

24. Akiba T., Sano S., Yanase T., Ohta T., Koyama M., Optuna: A next-generation hyperparameter optimization framework, [in:] *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ACM, Anchorage, AK, USA, 2019, pp. 2623–2631, https://doi.org/10.1145/3292500.3330701.

25. Lee S., Park S., Kim T., Lieu Q.X., Lee J., Damage quantification in truss structures by limited sensor-based surrogate model, *Applied Acoustics*, **172**: 107547, 2021, https://doi.org/10.1016/j.apacoust.2020.107547.

26. Torzoni M., Manzoni A., Mariani S., A multi-fidelity surrogate model for structural health monitoring exploiting model order reduction and artificial neural networks, *Mechanical Systems and Signal Processing*, **197**: 110376, 2023, https://doi.org/10.1016/j.ymssp.2023.110376.

27. Raiaan M.A.K. *et al.*, A systematic review of hyperparameter optimization techniques in convolutional neural networks, *Decision Analytics Journal*, **11**: 100470, 2024, https://doi.org/10.1016/j.dajour.2024.100470.

28. Hanifi S., Cammarono A., Zare-Behtash H., Advanced hyperparameter optimization of deep learning models for wind power prediction, *Renewable Energy*, **221**: 119700, 2024, https://doi.org/10.1016/j.renene.2023.119700.

29. Sacks J., Welch W.J., Mitchell T.J., Wynn H.P., Design and analysis of computer experiments, *Statistical Science*, **4**(4): 409–423, 1989, https://doi.org/10.1214/ss/1177012413.

30. Hutter M., The loss rank principle for model selection, [in:] N.H. Bshouty, C. Gentile [Eds], *Learning Theory. COLT 2007. Lecture Notes in Computer Science*, Vol. 4539, Springer, Berlin, Heidelberg, 2007, pp. 589–603, https://doi.org/10.1007/978-3-540-72927-3_42.

31. Wang B., Ding X., Wang F.-Y., Determination of polynomial degree in the regression of drug combinations, *IEEE/CAA Journal of Automatica Sinica*, **4**(1): 41–47, 2017, https://doi.org/10.1109/JAS.2017.7510319.

32. Kingma D.P., Ba L.J., Adam: A method for stochastic optimization, Paper presented at International Conference on Learning Representations 2015, San Diego, California, United States, 2015, https://doi.org/10.48550/arXiv.1412.6980.

Received November 18, 2025; revised April 24, 2026; accepted May 14, 2026; available online May 19, 2026; version of record July 3, 2026.