NORDDESIGN CONFERENCE
REYKJAVIK, ICELAND, 12-14 AUGUST 2024

Constrained Multi-Objective Design Optimization in Systems Engineering Using Active Learning

Oliver Bleisinger[1], Mareike Keil[2], Martin Eigner[3]

[1]University of Kaiserslautern-Landau, Germany
[2]University of Mannheim, Germany
[3]EIGNER engineering consult, Germany

Abstract: Design of complex systems demands exploration of high-dimensional design spaces due to competing design goals and numerous design parameters. Manual tradeoff studies in hyperspaces are laborious and hence, Artificial Intelligence offers a solution. This paper introduces a Constrained Multi-Objective Design Optimization approach in Systems Engineering using Artificial Intelligence and evaluates this method through an automotive case study. Therefore, an Active Learning algorithm is presented to automate identifying pareto fronts by inferring simulation models as substitution for learning data.

Keywords: Artificial Intelligence (AI), Machine Learning (ML), Systems Engineering (SE), Design Optimization, Simulation-Based Design

1 Problem Statement & Research Scope

System design and optimization in automotive Systems Engineering (SE) is heavily linked to well-established processes and approaches. A specific approach involves the usage of simulations, which are based on modeling of physical effect chains, to evaluate different architectural alternatives as well as to optimize design parameters. Various Machine Learning (ML) techniques enable the acceleration and significant improvement of simulation-based design optimization. To explore the potential of ML, several use cases are examined for the application of ML algorithms to specific simulation tasks in vehicle design. These investigations are carried out within the context of publicly funded research projects by the authors of this publication and associated industry partners. In this publication, the use case of solving Constrained Multi-Objective Design Optimization problems is closely investigated by implementing a prototype for an automotive case study considering early phases of SE. Figure 1 shows the localization of the research scope in a typical V-Model (left side) as well as the link to the standardized processes for testing (right side) according to the International Software Testing Qualifications Board (ISTQB), whereby the process steps are tailored to fit the research scope regarding simulation.

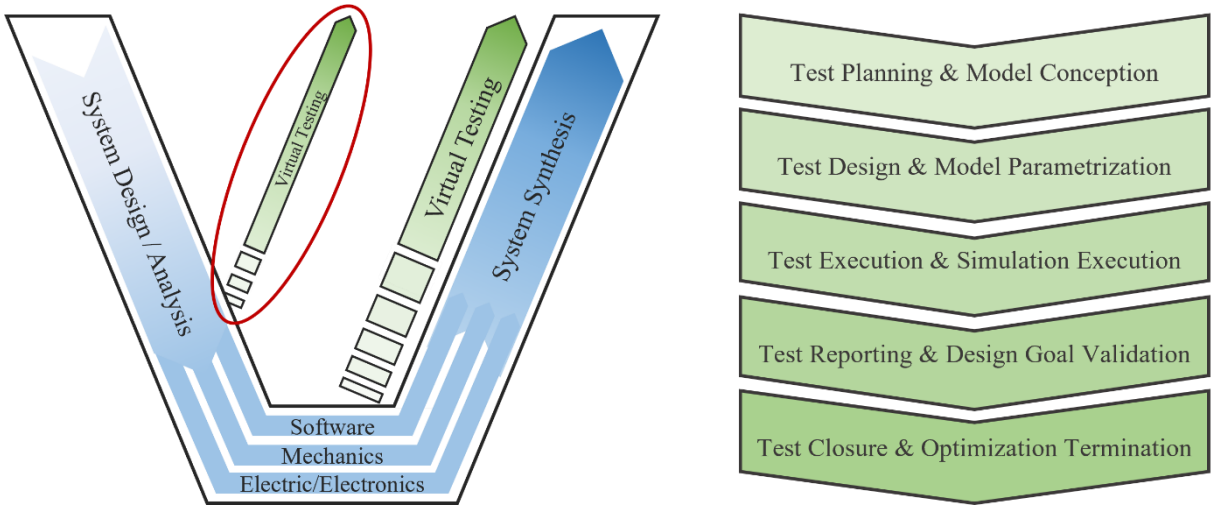

Figure 1. V-Model for Systems Engineering according to Eigner (Eigner et al., 2020) with localization of research scope (left) and testing process steps according to the International Software Testing Qualifications Board tailored for simulation (right).

Following research questions (RQ) are answered by implementing an Active Learning (AL) approach for the case study and exploring the potential of AI, specifically ML, to solve Multi-Objective Optimization (MOO) problems:

- RQ 1: Which support can AI provide for the optimization of competing system properties by automated design space exploration for complex and non-linear systems?

- RQ 2: How can AI and design knowledge from simulation models be combined to solve Constrained Multi-Objective Design Optimization problems in Systems Engineering applications?

Since a special class of MOOs is considered, this leaded to selection of the AL approach prior to this publication. As mentioned in RQ 2, a class of design problems is investigated, which requires fulfilment of design constraints and leads to the need to integrate these into the ML algorithm. AL enables native integration of constraints (Khatamsaz et al., 2023) and further ML algorithms, precisely Evolutionary Algorithms (EAs) as well as Soft and Asynchronous Advantage Actor-Critic algorithms, were investigated and evaluated beforehand. The presentation of the latter results is not part of this publication and comparison will be shown in a later publication, since currently no suitable metrics for comparison of different types of ML algorithms in real world applications exists. This is still ongoing research work of the authors.

Additionally, it is to be considered that the MOO case study (section 3) should be formulated as an inverse design optimization problem. Therefore, it is assumed that competing desired system properties are given, and specified values of design parameters should be found, which optimizes design tradeoffs according to the given system properties. Since a complex system is optimized in the case study, the MOO problem can be described in a hyperdimensional design space, which means that more than three design parameters should be determined. Additional to the research scoping by RQ 1/RQ 2, the following statements (STs) describe characteristics of the object of investigation more precisely:

- ST 1: The design problem considered is a MOO problem formulated as inverse design optimization problem for which four design goals and five design parameters will be given.
- ST 2: To solve the optimization problem the hyperdimensional design space should be automatically explored (design space exploration) by AL, taking (simplified) design constraints into account.
- ST 3: Design goals are competing in MOO problems, meaning that there is no single global optimum and instead parts of the pareto front will be described by the solutions found by the AI algorithm.
- ST 4: Simulation based on mathematical-physical behavior models should be used to evaluate the influence of variation of design parameters on desired system properties and to generate required data for application of ML.

2 Starting Point & State-of-the-Art

To clarify the current landscape of ML approaches solving MOOs, a survey of existing papers was conducted, mainly based on keywords related to design space exploration, ML, MOO and AL in concrete applications. It is important to note, that only the most important findings with relation to the problem statement in section 1 are presented. Consideration of general applications of AI in design as well as a broad overview of approaches to solve MOOs go beyond the scope of this paper. Furthermore, key terms related to ML algorithms are briefly given in this section.

The challenges addressed by MOOs are diverse and lead to different approaches to meet specific real world problem requirements. To narrow down relevant algorithms, two main key factors should be considered. One key distinguishing factor of different MOO approaches lies in types of pareto fronts that these methods can uncover, whereby a distinction is made between regular and irregular pareto fronts. Regular pareto fronts are always continuous, implying the design space to be continuous. Irregular pareto fronts are not necessarily continuous as for example in discrete design spaces spanned by discrete design parameters. The second distinguishing factor of MOO approaches is the classification of the leading concept to find optimal solutions within the design space. Mainly two types of MOO approaches are considered, namely so-called simulated annealing approaches and approaches based on EAs. For both classes of MOO approaches an exhaustive number of algorithms were developed over the last two decades and an overview can be taken from current publications (Saini and Saha, 2021) or standard literature (Russell and Norvig, 2021).

In relation to the investigated case study of this contribution, especially MOO approaches considering a discrete pareto front and based on EAs seem best suitable. EAs like NSGA-II (Non-dominated Sorting Genetic Algorithm II) (Deb et al., 2002) can identify multiple optimal solutions of the pareto front in one run and do not get stuck in local optima in early runs. This is important since design goals are competing, meaning there is not a single globally optimal design and instead the task of optimization can either be described as finding multiple optimal tradeoffs within the design space or finding an optimum fulfilling a predefined cost function. In the latter case it might be difficult to formulate the necessary cost function in such a way that, e.g. the weighing of the competing design goals, leads to a suitable solution. Furthermore, MOO approaches based on EAs seem suitable since they have the capability to approximate solutions in hyperdimensional design spaces while striking a balance between runtime and solution accuracy (Chugh et al., 2019). Still, main drawbacks of EAs are the missing native consideration of design constraints while exploring the design space and the inability to consider new insights, either from human feedback or new data at runtime of the optimization. While the latter requirement can be fulfilled by Reinforcement Learning (RL) approaches as well and was investigated on the example of the Soft Actor-Critic (SAC) algorithm and Asynchronous Advantage Actor-Critic (A3C) by the authors, consideration of design constraints by native design of the algorithm is not possible in the same way as with AL algorithms.

AL can briefly be described as ML approach or paradigm in which not a fixed dataset is considered, but rather the algorithm requests specific data points to improve its performance. Understanding the full potential of AL for the case study involves recognizing that the MOO problem and AL architecture presented in this paper utilizes design knowledge captured in a simulation model, as addressed by RQ 2. Therefore, the key idea using AL is to enable the ML algorithm to actively prompt the simulation model and extract the most valuable design knowledge for the optimization process. This can be considered a more efficient way instead of relying on RL algorithms or going with MOO approaches based on EAs. Therefore, this paper considers the usage of AL, which is novel for the concrete type of MOO in respect to the case study. Classically, AL is used in ML applications related to classification and regression tasks as well as natural language processing. Further details for AL in concrete applications can be found in publications (Cevik et al., 2015).

Additionally, there is a recognizable trend towards using methods based on Artificial Neural Networks (ANN) or EAs in engineering. For example, Hong and Le (Hong and Le, 2023) use an ANN-based method for the optimized design of doubly reinforced rectangular concrete beams based on multi-objective functions. Ebbs-Picken et al. (Ebbs-Picken et al., 2023) point out that the most advanced methods for MOOs of battery thermal management system design involves the use of genetic algorithms. Caruana and Lou (Caruana and Lou, 2021) show that methods based on swarm intelligence, which count as evolutionary methods, offer possibilities for solving various MOO problems. There are also some further applications using ML for design space exploration, of which the most important are briefly outlined below.

- Search and optimization of the position of FPGAs in E/E development (Murray and Betz, 2019)
- Disease classification and tradeoffs to reduce medication side effects (Yu et al., 2019)
- Solving inverse optimization problems to optimize complex optical systems (Wankerl et al., 2020)
- Reduction of the design space through clustering of design parameters (Qiu et al., 2016)

To distinguish this contribution from the mentioned work, it is important to note that most of them focus on design space exploration that either does not optimize competing parameters (no MOO), requires pre-preparation of the design space, or does not yield specific combinations of design parameters as output. In further publications, the use of EAs rather than AL has mostly been considered or inverse optimization problems have not been solved – meaning the specification of desired system properties and the search for design parameters that optimize these system properties was not examined. In summary, the state-of-the-art misses to utilizing AL to solve design problems depicted by section 1 of this contribution.

3 Considered Case Study

The case study focuses on system design and optimization with computer simulations in an automotive development process. Specifically, the study aims to support the parametrization of key components in vehicle design through tradeoff studies to optimize competing system properties. Thus, the case study is focused on vehicle conception in SE and not the optimization of individual components. This requires considering cross-component causal effect chains, leading to emergent behavior not solely predictable based on the component behavior according to the theory of complex systems (Thurner et al., 2018). Furthermore, not only behavior of one engineering domain, as mechanical or electrical engineering, should be considered, but also software-based behavior. To generate data and enable the AL algorithm to actively query feedback (see section 4) to learn and adapt to the system's behavior, a simulation model is set up. For reasons of publication the simulation model depicts a classical car with internal combustion engine, but the concepts are transferable to any kind of vehicle design. The simulation model is also required not only for the AL approach but also because the vehicle's behavior can't be captured by a single formula or closed system of equations. Hence, the system's behavior, as described by a longitudinal vehicle behavior model, is non-linear and not easily optimized without computer simulations. The utilization of simulations for optimizing design parameters often follows the scheme depicted on the right side of Figure 2.

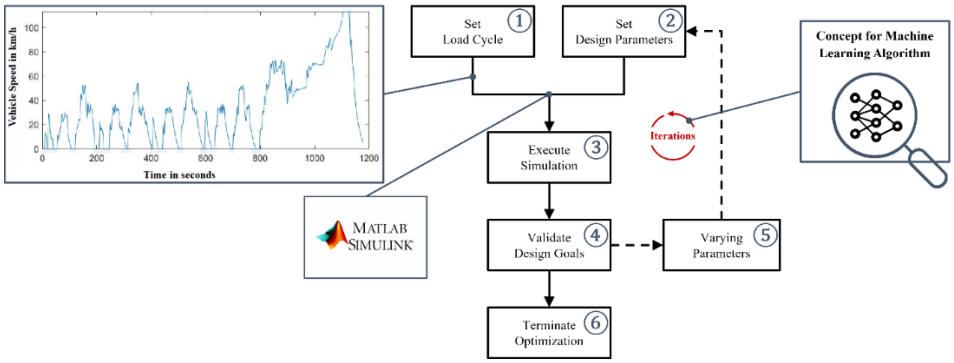

Figure 2. Process of design optimization through simulation according to Bleisinger et al. (Bleisinger et al., 2022) and representation of the scope of Machine Learning based automation for iterative parameter variation and simulation.

The depicted process of the simulation-based optimization with the steps 1 to 6 in Figure 2 is a more detailed view on the virtual testing approach already presented in Figure 1 (right side). Therefore, this can be understood as specialization of the generic testing process defined by the International Software Testing Qualifications Board by tailoring these for virtual testing by computer simulations. The diagram illustrates that, for a specific load cycle (step 1) and selected design parameters (step 2), simulations are conducted (step 3) to determine potential system properties. The system properties are typically validated manually by simulation engineers (step 4). If the design goals are not met or optimization is still possible, the design parameters are varied (step 5) before another iterative loop through steps 1 to 5 is carried out with the new design parameters and another evaluation of the design goals is conducted. The optimization is terminated if required design goals are met (step 6). The main task in conception of the ML algorithm is to automize steps 2 to 6. This means the whole process of design optimization is considered in the case study and only the load cycles are stable through the iterations for optimization after the initial setup. At this point, it should be mentioned that the main load cycle of the longitudinal vehicle model is represented by measurement data (Figure 2, top left) from a roller test stand, collected by one of the authors in a laboratory and emulating the New European Driving Cycle (NEDC). Furthermore, a second load cycle is needed to generate simulated output data according to the design goals in Table 1. The second load cycle is therefore a full throttle scenario at maximum acceleration of the simulated vehicle.

The simulation model was built in Matlab/Simulink, integrated in the AL architecture according to Figure 4 and receives as input/stimuli the two mentioned load cycles as well as five design parameters. In the performed case study, energy consumption and different indicators for driving performance are considered as design goals. The input parameters for the simulation, notably representing the design parameters in the MOO problem, include the axle ratio, tire rolling radius (reflecting tire size) and the gear ratios of gears three to five of the step transmission. In summary, the following results of the simulation model should be considered as system properties, along with the following design parameters:

Table 1. Summary of design parameters and design goals (desired system properties) for the Multi-Objective Optimization problem.

Design Parameter	Description
i_{ax}	ratio of the axle transmission
r_{tr}	rolling radius of the tires
i_{g3}	ratio gear 3 of step transmission
i_{g4}	ratio gear 4 of step transmission
i_{g5}	ratio gear 5 of step transmission

Design Goals	Description
fc	fuel consumption for load cycle
el_{g3}	elasticity value in gear 3
el_{g4}	elasticity value in gear 4
el_{g5}	elasticity value in gear 5

The objective is to optimize the four system properties according to a specified scoring function and simultaneously satisfying formulated design constraints. One of the four system properties considered as design goals in the MOO is the overall fuel consumption required by the vehicle concept to run through the defined load cycle, represented by the speed profile in Figure 2 (top left). Additionally, three driving performance indicators are considered as design goals, specifically, the elasticity values for full-throttle acceleration from 60km/h to 100km/h. The elasticity values are given in seconds necessary for acceleration and are calculated using the full-throttle load cycle in the simulation model. To simplify the design constraints for explainability in the case study, fixed value ranges for all design parameters are used as constraints, but the approach presented is transferable to more complex design constraints, e.g. based on mechanical boundaries for construction and geometric integrity of cogwheels for the step transmission. To ensure the readers overview, the capabilities of the Matlab/Simulink simulation model is described as input/output relation (blackbox) in Figure 3 (right side) and a depiction of the main idea for automation of steps 2 to 6 by AL (Figure 3, left side) is provided. For further details on the simulation of energy consumption and performance indicators, the authors refer the interested reader to additional literature (Knöbl, 2023).

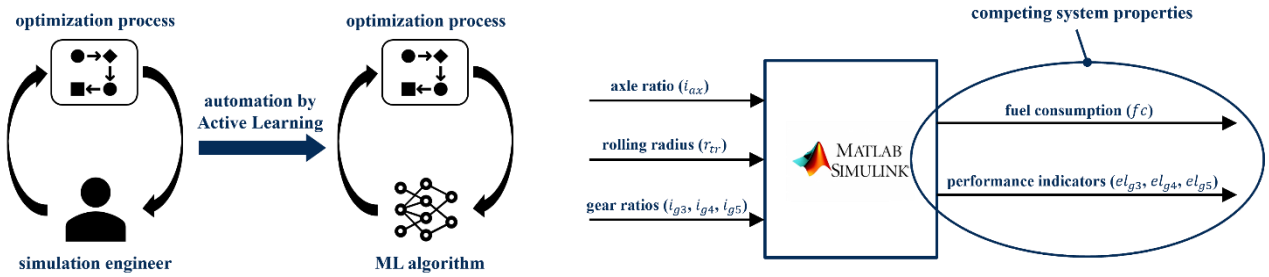

Figure 3. Depiction of automation of iterative parameter variation, simulation, evaluation (left side) and depiction of input/output relations for the Matlab/Simulink model of longitudinal vehicle dynamics behavior (right side).

4. Active Learning Implementation

AL is an iterative learning method that trains ANNs on an iteratively extended labeled dataset by choosing the most informative examples from unlabeled data and prompting labeling feedback either from humans or newly generated data at runtime of the algorithm. Labeling in the presented application means, that sets of design parameters are scored in respect to fulfilment of the design goals by evaluating a scoring function based on the output of the simulation model and design goals. Furthermore, AL utilizes a score prediction by a trained ensemble of ANNs to identify the most informative candidates of design parameter to be scored and be added to the extended labeled dataset. In general, different algorithms exist to identify the most informative candidates of design parameters as e.g. evaluating the uncertainty of the predicted score and deriving necessity for ground truth because of high level of uncertainty in the prediction. For the presented procedure in this paper, it is important to recognize that AL might be a type of ML approach but is not necessarily associated with a high demand on existing data. It is rather the case, that the required learning data is drastically reduced by the AL algorithm so it can solely rely on actively prompting the most useful data from a single simulation model.

Dependent on the concrete setup of the algorithm, modern AL architectures can be considered rather complex, although the concept of AL is already well-known since the early 1980s (Russell and Novig, 2021). Originally, AL was considered suitable to iteratively learn by querying feedback from a human teacher (Ren et al., 2020). According to RQ 2, in the case study not human feedback, but feedback from a simulation model representing human design knowledge is investigated in this publication. AL is a valuable approach nowadays, used in various applications where labeled data is scarce or expensive to obtain, since it is possible that the algorithm intelligently chooses the most informative instances for labeling, reducing the number of simulations in the case study. Since the mentioned ensemble of ANNs (see section 4.2), which predict scores and select the most informative design parameter combinations, is trained based on iteratively labeled data, the AL approach can be considered a supervised ML method.

Because the overall AL architecture in the case study might be unfamiliar to readers, the following depiction is provided as overview and allocations to process steps from Figure 2 (right side) are provided. Furthermore, in the following subsections the architectural components of the AL approach are briefly described, including the utilized algorithm design.

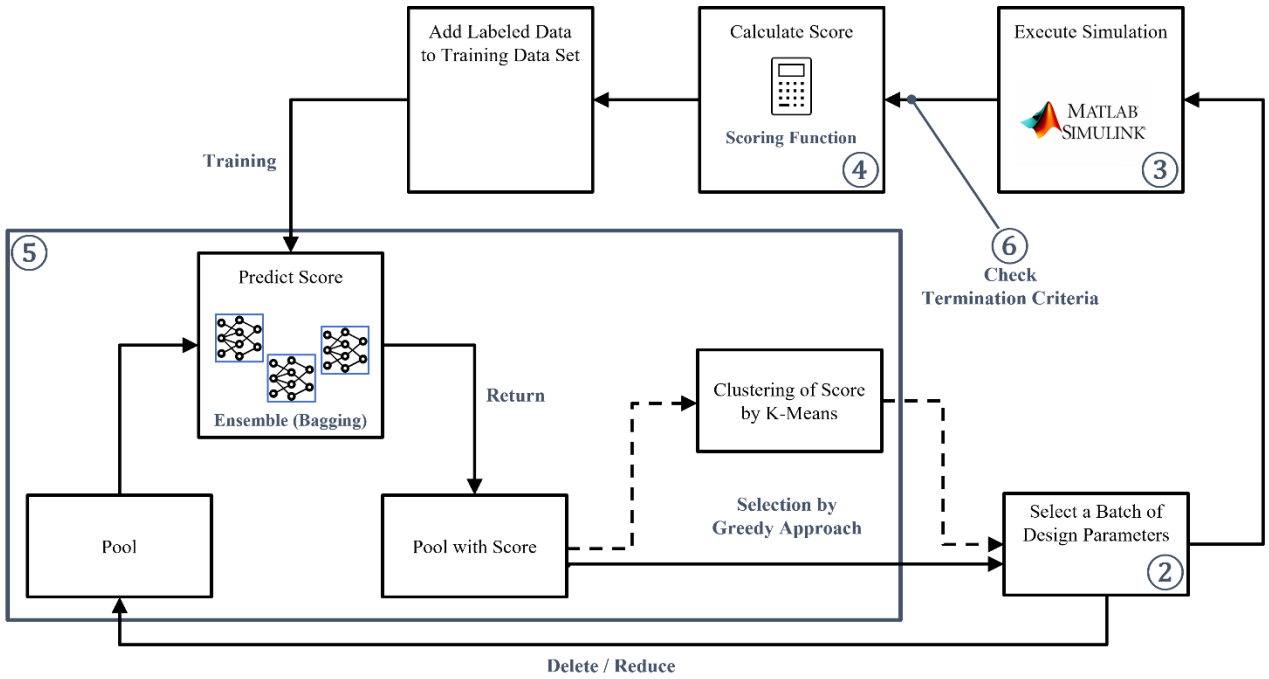

Figure 4. Architectural design of the AL algorithm to automate iterative parameter variation and simulation in design optimization.

4.1 Pool Setup

The algorithm operates on discretized parameters within the hyperdimensional design space. A selected design parameter combination of the sum of all possible design parameter combinations is potential input for the simulation model. The discretization is mainly defined by the parameters and constraints shown in Table 2, although it should be noted, that the discretization is performed very roughly (high step width) to reduce execution time of the AL approach for reasons of demonstration in this publication. This means that the presented AL approach can easily be adjusted to handle a finer step width for discretization in exchange for a significant higher computational effort and time consumption to solve the MOO problem. Moreover, it is worth mentioning that the lower and upper bounds for discretization as well as for the step width is selected based on estimations for the technical feasibility of design parameters for the considered vehicle concept.

Table 2. Discretization of the design space with lower bounds, upper bounds and step widths for design parameters.

Design Parameter	Lower Bound	Upper Bound	Step Width
i_{ax}	3.00	5.50	0.250
r_{tr}	0.20	0.40	0.100
i_{g3}	0.50	2.25	0.125
i_{g4}	0.50	2.25	0.125
i_{g5}	0.50	2.25	0.125

By applying the discretization shown in Table 2, a total number of design parameter combinations of 54,880 within the hyperdimension can be defined. At this point, it must additionally be considered, that there is another constraint in the MOO, which is the monotonic order of gear ratios easily described by Equation (1).

$$i_{g3} > i_{g4} > i_{g5} \quad (1)$$

Yet, the mentioned set of 54,880 design parameter combinations must be reduced by sets disregarding the constraint of monotonic gear ratio values. To address this, parameter combinations failing to meet the condition are removed from the initial pool, which leads to a reduced pool size of 7,280 design parameter combinations. This refined dataset constitutes the pool, which should be referred to as X_{pool}. Even if the pool size seems quite small, the approach for setup of X_{pool} can be applied to any constrained MOO problem since every constraint can be expressed as a logical formula that can be either satisfied or unsatisfied by parameter combinations. This greatly distinguishes the AL approach from a RL approach using a reward function to ensure that constraints on design parameters are met. Furthermore, it should be noted, that for the AL approach, constraints on the output parameters of the simulation model can be met by defining the scoring function accordingly (see section 4.3). In RL approaches both, the constraints on design parameters and output parameters of the simulation model, must be handled by the reward function so the design space is not reduced prior to optimization runs, potentially making RL less efficient for highly constrained MOOs.

4.2 Ensemble Definition

In past research, AL was already implemented for calibration of a model for diagnosis of breast cancer (Cevik et al., 2015). In the context of the AI use cases researched by the authors, this correlates to "automatized parameterization of simulation models". Hence, this is out of scope for the research scope presented in section 1, which can be roughly condensed to the AI use case "solving Multi-Objective Optimization problems" and is therefore not investigated in this contribution. But although the application of AL for solving MOO problems was not considered in the publication of Cevik et al., the proposed framework for AL is adapted for the case study presented in this publication. Therefore, the pool-based AL algorithm should be enhanced by an ensemble of neural networks as proposed in the architecture in Figure 4 to enable the predicting of scores for design parameter combinations which can be expressed as a regression task: predicting a continuous scalar score based on continuous numeric input parameters.

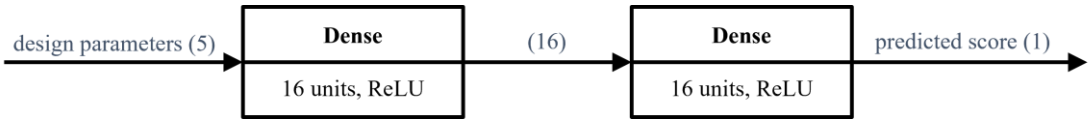

Figure 5. Base architecture for neural networks of the ensemble of the Active Learning approach.

The ensemble, comprising a defined number of ANNs with similar neural architecture, should employ simple multi-layer perceptrons. In total five ANNs according to hyperparameters depicted in Figure 5 are utilized, meaning that the input layer consists of the same number of neurons as the quantity of design parameters and the output layer's number of neurons is exactly one and suits the dimension of the scoring which is a scalar value. Furthermore, the ANNs contain two hidden layers with 16 neuron per hidden layer in which a ReLU activation function is utilized. The adam optimizer is selected for training with a learning rate $\lambda = 0.01$ for each ANN. To calculate the training loss the mean squared error (MSE) is applied and the ensemble is evaluated as a whole by using the mean absolute error (MAE). Each ANN is initialized with different weights and trained with distinct data, employing bootstrap aggregation (bagging). Additionally, it is important to state, that the evaluation dataset (D_E) and training dataset (D_T) are formed by random equally distributed sampling from the so far labeled dataset. For training of individual ANNs of the ensemble, D_T is further divided into D_{train} and D_{test}, with D_{test} created by repeatedly drawing random samples with replacement from D_T (bootstraps) as stated in Equation (2).

$$D_{test} = D_T - D_{train} \quad (2)$$

As AL uses small datasets, the bagging method results in individual training datasets ($D_{train,1}$, …, $D_{train,5}$), allowing data duplications during the sampling process and introducing bias, which impacts the trained ANNs in the ensemble. The final prediction is therefore derived from the mean value of individual biased predictions, often yielding higher accuracy than a single ANN network (Breiman, 1996). The evaluation dataset D_E exploits overall ensemble performance by evaluating the final prediction, while D_{test} assesses the performance of a single ANN within the ensemble during training. Since in the AL architecture according to Figure 4 K-Means is used for clustering, a default value of $B = 10$ is used for the number of clusters. In addition, the size of the initial dataset is by default chosen as $D_{init} = 10$.

4.3 Scoring Method

By varying the design parameters, the simulation model behaves differently with respect to the fuel consumption and the driving performance. Balancing the conflicting design goals of minimizing fuel consumption and maximizing driving performance requires a tradeoff, making the AL approach aim for a pareto-optimal solution and in need of a sophisticated scoring function. For the case study a so-called greedy approach for AL among various options for scoring a sample design parameter combination is chosen. The scoring method assesses a sample's quality with respect to the design goals. Using the greedy approach for the MOO, locally optimal choices at each epoch are chosen hoping to find a global optimum over time. This means, each epoch/iteration of the AL process, the greedy approach selects the best available option based on the scoring. Therefore, the scoring function must be able to determine the optimality of parameters within the solution space which is spanned by the possible value ranges of the design goal values (not to be confused with the design space).

The used scoring method is primarily based on the euclidean distance, aiming to determine its global optimum. It is important to note, that the optimal system properties cannot be precisely estimated before performing the MOO, but it is not necessary for the scoring function to know the real global optimum. Instead, a roughly estimated optimum s^* or rather a fictive optimum in the middle of the design goal ranges for the possible system properties can be selected for calculation of the euclidean distance. However, a drawback arises due to the uneven distribution of the design goal ranges, where different-sized intervals are defined by the individual design goals depicted in Table 3.

Table 3. Minimum and maximum values for the design goal ranges to be optimized.

Design Goal	Hypothetic Minimum	Hypothetic Maximum
fc	1	1.7
el_{g3}	1	10
el_{g4}	1	10
el_{g5}	1	12

The solution space, spanned by two design goal's value ranges forms a rectangle. The issue lies in the euclidean distance scoring, which distributes scores radially. In the context of AL, a vector of the design variables $i_{ax}, r_{tr}, i_{g3}, i_{g4}, i_{g5}$ is used for the design parameter combinations and the required scoring function must include the simulation model's output to be able to clearly score and calculate possible domination of design parameters. With respect to the mentioned requirements, following scoring function can be defined:

$$f_{score}(x) = f_{score}(i_{ax}, r_{tr}, i_{g3}, i_{g4}, i_{g5}) = \frac{1}{\|f_{sim}(x) - s^* \circ \hat{s}\|} \quad (3)$$

and

$$f_{score}(x) = 10^3 \text{ if } f_{sim}(x) = s^* \quad (4)$$

where $\circ$ represents the element-wise product of two vectors, f_{sim} represents the result/output of the simulation, $s^* = < 1.35;\ 5.5;\ 5.5;\ 6.5 >$ and $\hat{s} = < 0.7^{-1};\ 9^{-1}; 9^{-1}; 11^{-1} >$

4.4 Termination Criteria

The termination criterion varies across concrete applications for AL. As an example (Saadallah et al., 2018) uses a fixed number of simulation calls and thus the sampled labeled dataset as termination criterion, neglecting the quality of the sampled dataset. This may lead to a suboptimal selection of design parameters. In the breast cancer detection application (Cevik et al., 2015), termination occurs after a fixed number of iterations without finding a solution, but this method suffers from potentially overlooking progress during the search in a low-scoring region. In contrast, an engine design application (Owoyele and Pal, 2020) employs two criteria, evaluating exploration and exploitation stagnation. This approach considers differences in sampled points and fitness value improvement. While the risk of early termination exists, it utilizes more information about search progress. Since the application of this paper is to present general feasibility of the AL approach

without optimizing runtime properties of the AL algorithm, a simple termination criterion is chosen. Therefore, a simple property of the AL algorithm should be used and so the termination criterion is set to 100 iterations. This also enables comparison of the performance for different hyperparameter configuration in later studies.

4.5 Algorithm Design

The AL algorithm aims to identify valuable design parameter combinations without simulating all design parameter combinations in the pool X_{pool}. It should be noted, that it wouldn't be of value, if all possible combinations of design parameters in X_{pool} would need to be computed by the simulation model, because this would lead to the worst-case scenario in runtime, leading to a brute force exploration of the design space. To make efficient exploration of the design space possible, the ensemble is initially trained with a small randomly selected dataset X_{init} from the pool X_{pool}. Therefore, a label must be assigned to the samples in X_{init}. To do this, f_{score} is evaluated by running the simulation model and the respective ground truth score is assigned as a label to the considered set of design parameter combinations X_{init}. The sampled data is removed from the pool so that $X_{pool} \leftarrow X_{pool} - X_{init}$ holds after the initial training of the ensemble.

After this initial training, the ensemble scans through the pool X_{pool} and determines for each instance $x \in X_{pool}$ the predicted score $\hat{f}_{score}(x) = \hat{y}$. Subsequently, the samples in X_{pool} are ranked in descending order by $\hat{y}$ and the best 80 instances are used for a clustering procedure as proposed by Cevik et al. (Cevik et al., 2015). For the clustering a K-Means algorithm is used, where K depicts the number of clusters, which is chosen to be 10 based on several experiments. From each cluster, the best sample x_i with the highest score $\hat{y}$ is selected. In addition, a constant number c_{rand} of randomly drawn samples X_{rand} is determined and added to a sampled set. For the sampled set $\{x_1, \dots, x_K\} \cup X_{rand}$, the true score y is determined by evaluating f_{score}, which means the simulation is run for the whole sampled set and the true scores are calculated by utilization of the score function. The true scores are then assigned as labels to the individuals of the sampled set and the labeled design parameter combinations are then added to the training data. The sampled data are furthermore removed from the pool X_{pool} in this process. Finally, a new iteration of the AL algorithm begins, starting with re-training the ensemble on the extended training dataset. A limit of simulation calls is set as a termination criterion for the greedy algorithm as described above, also meaning that the algorithm ends as soon as the number of labeled data exceeds D_{max}. The whole algorithm is additionally depicted as pseudocode in Figure 6. (Stuppy, 2023)

Initialize $|D|_{\max}, K, c_{rand}$
Randomly select $X_{init} \subset X_{pool}$
$Y_{init} \leftarrow f_{score}(X_{init})$
$D \leftarrow (X_{init}, Y_{init})$
$X_{pool} \leftarrow X_{pool} - X_{init}$
while number of labeled data $< |D|_{\max}$ **do**
    $\hat{f}_{score} \leftarrow$ Fit ensemble on training data D
    Predict scores $\hat{f}_{score}(X_{pool})$
    $X_{ranked} \leftarrow$ Ranking instances X_{pool} by $\hat{f}_{score}(X_{pool})$ in descending order
    $C_1, ..., C_K \leftarrow$ Perform K-Means clustering on $X_{ranked}[: 80]$
    $x_1, ..., x_K \leftarrow$ Select the highest scored instance from each cluster
    $X_{rand} \leftarrow$ Randomly draw c_{rand} instances
    $Y \leftarrow f_{score}(\{x_1, ...x_K\} \cup X_{rand})$
    $D \leftarrow D \cup (\{x_1, ...x_K\} \cup X_{rand}, Y)$
    $X_{pool} \leftarrow X_{pool} - (\{x_1, ...x_K\} \cup X_{rand})$
    if termination criterion reached **then**
        break
    end if
end while

Figure 6. Pseudocode of the implemented greedy Active Learning approach.

5. Exemplary Results

To evaluate the results of the presented AL approach for solving the described MOO problem, the metric is defined as number of valid solutions for the case study. A solution or combination of design parameters is considered valid, if it fulfils certain constraints regarding the design goals, which are not presented in this paper for the sake of clarity and conciseness. Therefore, to be more specific regarding the metric, the cumulative number of valid solutions per iteration was considered over the termination criteria of 100 iterations. In addition, the algorithm was tested with 10 different seeds, as there are many stochastic elements in the algorithm that can influence performance. Preselection of seeds allows reproducible of the following statistical influence which are not an exhausting list, but rather examples of main influences:

- the random division of the data into training and test dataset
- the random determination of the initial dataset that is labeled
- the initialization of the weights of the neural networks

The cumulative number of valid design parameter solution combinations found, both evaluating the average and the median, was used as the measure for evaluation across the execution of the ten seeds. Furthermore, the AL architecture and algorithm was tested with several sets of hyperparameters, although it could be observed that the results were quite similar in performance, even when the hyperparameters of the ANNs in the ensemble were varied in a meaningful way. Nevertheless, the best hyperparameters were selected for the following presentation of results and are chosen as depicted in section 4.2. Finally Figure 7 shows the results with (left) and without (right) K-Means clustering as an intermediate step in the AL architecture according to Figure 4. It can be observed that there is little difference when utilizing the clustering. In conclusion, clustering can therefore also be omitted to gain more performance regarding computational effort.

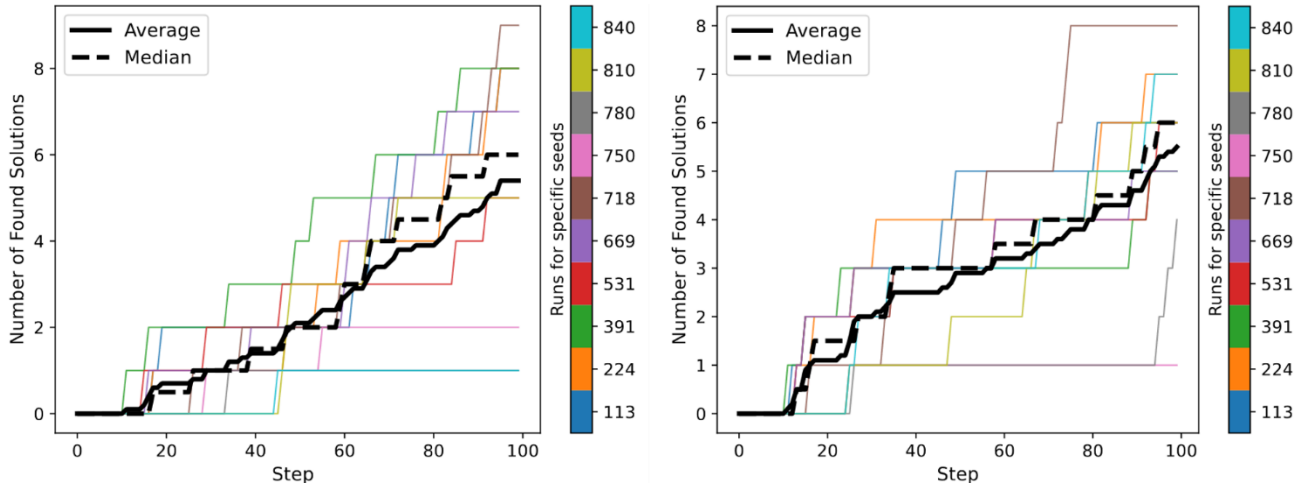

Figure 7. Depiction of found valid solutions with AL for optimal combinations of design parameters (left: with clustering; right: without clustering; colors depict results for different seeds).

While Figure 7 is just presenting the number of valid solutions per run of the AL algorithm for different seeds, Table 4 shows what the concrete solutions look like. The solutions are presented for the AL architecture without the additional application of the K-means clustering and consists of the found combinations of design parameters, the corresponding values of the design goal variables determined by the simulation model and the result of the scoring function. In summary, it can be stated, that the solutions for the MOO problem found by the AL approach are feasible designs on the pareto front.

Table 4. System properties and design parameters for AI generated system designs.

i_{g3}	i_{g4}	i_{g5}	i_{ax}	r_{tr}	fc	el_{g3}	el_{g4}	el_{g5}	f_{score}
2.00	1.75	1.50	3.00	0.25	1.610783	6.60	8.69	11.83	0.717039
2.00	1.75	1.50	4.25	0.35	1.621922	6.43	8.49	11.55	0.694777
1.75	1.50	1.25	3.00	0.20	1.692209	5.44	7.56	10.90	0.671883
2.00	1.75	1.50	3.75	0.30	1.650784	6.03	8.01	10.90	0.652596
2.00	1.75	1.50	4.50	0.35	1.681201	5.68	7.56	10.31	0.629780
2.00	1.75	1.50	3.25	0.25	1.693895	5.55	7.39	10.08	0.625624

6. Conclusion & Future Work

In conclusion, the effective application of Machine Learning (ML), especially Active Learning (AL), for Multi-Objective Optimization (MOO) problems in design optimization is based on meeting prerequisites. Once these criteria are fulfilled, the applicability of ML to design optimization is possible. The most essential requirements include:

- Knowledge of possible value ranges for design parameters to be chosen
- Definition of target ranges for design goals, which indicates valid solution of the design problem
- Establishment of a scoring function to be embedded in the Active Learning approach

Considering the second research question (RQ 2) from section 1 it can be stated that AL notably facilitates the straightforward consideration of design constraints in MOOs. Therefore, Constrained MOO problems can be assisted by AL. With respect to the aspect of integration of design knowledge from simulation models with ML it can be stated clearly and explicitly that combination is possible and leads to significant possibilities to automate different activities in design optimization. The presented AL architecture in this paper provides a first answer how this can be implemented in technical means and therefore answers RQ 2. It should be mentioned that only the AL approach is investigated in this paper and conclusions are not transferable to other types of ML approaches as e.g. constraints are not always supported natively by different ML algorithms. This will be further investigated in future work and a metric will be developed to classify suitability of ML algorithms for Constrained MOOs and additional algorithms will be implemented.

With respect to the first research question (RQ 1) and limited to the investigated case study, it can be said, that design optimization of competing design goals or rather system properties is possible by application of ML. In the case study a complex and non-linear system was the object of investigation and ML or at least AL can be considered a possibility to automatically explore design spaces if specific conditions of the design space hold. As part of additional questions for future work, exploring transfer learning methodologies is identified as a promising avenue for further enhancing the capabilities of AL in the context of design optimization. This approach has the potential to leverage knowledge gained from one optimization task to improve the efficiency and effectiveness of applications with certain level of similarity.

References

Bleisinger, O., Eigner, M., Kuhn, T., 2020. Autogeneration of simulations model from usage data using learning AI procedures. ProductData Journal, vol. 2020-1, pp. 47-53. link: https://prostep.epaper-pro.org/pdj_1-2020_english/#48.

Bleisinger, O., Malek, C., Holbach, S., 2022. Machine Learning Based Simulation for Design Space Exploration. Proceedings of the Design Society, pp. 1521-1530. doi: https://doi.org/10.1017/pds.2022.154.

Breiman, L., 1996. Bagging predictors. Machine Learning 24, pp. 123–140. doi: https://doi.org/10.1007/BF00058655.

Caruana, R., Lou, Y., 2021. A Survey on Multi Objective Optimization Challenges in Swarm Intelligence. Journal of Computing and Natural Science, pp. 121-129. doi: https://doi.org/10.53759/181X/JCNS202101018.

Cevik, M., Ergun, A., Stout, N., Trentham-Dietz, A., Craven, M., Alagoz, O., 2015. Using active learning for speeding up calibration in simulation models. Medical decision making, 36, 10. doi: https://doi.org/10.1177/0272989X15611359.

Chugh, T., Sindhya, K., Hakanen, J. et al., 2019. A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. Soft Comput 23, pp. 3137–3166. doi: https://doi.org/10.1007/s00500-017-2965-0.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, vol. 6, no. 2, pp. 182-197. doi: https://doi.org/10.1109/4235.996017.

Ebbs-Picken, T., da Silva, C. M., Amon, C. H., 2023. Design optimization methodologies applied to battery thermal management systems: A review. Journal of Energy Storage, 67, 107460. doi: https://doi.org/10.1016/J.EST.2023.107460.

Hong, W. K., Le, T. A., 2023. ANN-based optimized design of doubly reinforced rectangular concrete beams based on multi-objective functions. Journal of Asian Architecture Engineering, pp. 1413-1429, doi.: https://doi.org/10.1080/13467581.2022.2085720.

Khatamsaz, D., Vela, B., Singh, P. et al., 2023. Bayesian optimization with active learning of design constraints using an entropy-based approach. npj Computational Materials 9, 49. doi: https://doi.org/10.1038/s41524-023-01006-7.

Knöbl, M., 2023. Vehicle modelling for the simulation of real-world-driving energy consumption and CO2-emissions. Master thesis, University of Wien. doi: https://doi.org/10.34726/hss.2023.99002.

Murray, K. E., Betz, V., 2019. Adaptive FPGA Placement Optimization via Reinforcement Learning. ACM/IEEE 1st Workshop on Machine Learning for CAD (MLCAD), pp. 1-6. doi: https://doi.org/10.1109/MLCAD48534.2019.9142079.

Owoyele, O., Pal, P., 2020. A novel machine learning-based optimization algorithm (ActivO) for accelerating simulation-driven engine design. Applied Energy, vol. 285. doi: https://doi.org/10.48550/arXiv.2012.04649.

Qiu, H., Xu, Y., Gao, L., Li, X., Chi, L., 2016. Multi-stage design space reduction and metamodeling optimization method based on self-organizing maps and fuzzy clustering. Expert Syst. Appl., 46, pp. 180-195. doi: https://doi.org/10.1016/j.eswa.2015.10.033.

Ren, P., Xiao, Y., Chang, X., Huang, P., Li, Z., Chen, X., Wang, X., 2020. A Survey of Deep Active Learning. ACM Computing Surveys (CSUR), 54, pp. 1 - 40. doi: https://doi.org/10.48550/arXiv.2009.00236.

Russell, S., Norvig, P., 2021. Artificial Intelligence: A Modern Approach. 4th Edition, Pearson Education. ISBN: 9780134610993.

Saadallah, A., Finkeldey, F., Morik, K., Wiederkehr, P., 2018. Manufacturing Systems Stability prediction in milling processes using a simulation-based Machine Learning approach. 51st CIRP Conference on Manufacturing Systems (CIRP CMS 2018), 72, pp. 1493–1498. doi: https://doi.org/10.1016/j.procir.2018.03.062.

Saini, N., Saha, S., 2021. Multi-objective optimization techniques: a survey of the state-of-the-art and applications. Eur. Phys. J. Spec. Top. 230, pp. 2319–2335. doi: https://doi.org/10.1140/epjs/s11734-021-00206-w.

Stuppy, L., 2023. Conception and Evaluation of a Reinforcement Learning Algorithm for Autoparametrization of Simulation Models. Bachelor thesis, University of Kaiserslautern-Landau.

Thurner, S., Klimek, P., Hanel, R., 2018. Introduction to the Theory of Complex Systems. Oxford Scholarship Online. doi: https://doi.org/10.1093/oso/9780198821939.001.0001.

Wankerl, H., Stern, M. L., Mahdavi, A., Eichler, C., Lang, E. W., 2020. Parameterized reinforcement learning for optical system optimization. Journal of Physics D: Applied Physics, 54. doi: https://doi.org/10.1088/1361-6463/abfddb.

Yu, C., Liu, J., Nemati, S., 2019. Reinforcement Learning in Healthcare: A Survey. ACM Computing Surveys (CSUR), 55, pp. 1 - 36. doi: https://doi.org/10.1145/3477600.

Contact: O. Bleisinger, University of Kaiserslautern-Landau, Department of Mechanical and Process Engineering, Gottlieb-Daimler-Straße, 67663 Kaiserslautern, Germany, bleising@rptu.de