

Metamodel-based Simulation Optimization Using Machine Learning for Solving Production Planning Problems in the Automotive Industry

Felicia Schweitzer
Mercedes-Benz AG Stuttgart
felicia.schweitzer@mercedes-benz.com

Lars Habel
Mercedes-Benz AG Stuttgart
lars-christian.habel@mercedes-benz.com

Oscar Jhonny Canaviri Vilca
Mercedes-Benz AG
Sindelfingen
oscar_jhonny.canaviri_vilca@mercedes-benz.com

Tobias Kulzer
Mercedes-Benz AG Sindelfingen
tobias.t.kulzer@mercedes-benz.com

Sigrid Wenzel
University of Kassel
s.wenzel@uni-kassel.de

Abstract

Due to the rising complexity of production systems in the automotive industry, simulation has become an established tool for analyzing dynamic systems. However, once the number of parameter combinations rises exponentially, the generation and evaluation of all possible solutions gets impractical. While the combination of simulation and optimization has a long tradition in academic research, its adoption in the automotive industry remains limited, often due to the high execution time associated with optimization experiments. To enable more efficient decision-making, this paper explores the integration of machine learning and optimization for simulation optimization. Specifically, it focuses on the use of metamodels incorporating various machine learning algorithms and metaheuristics to optimize two production planning problems with multiple parameter classes. The presented approach enables decision-makers to conduct a rapid assessment of complex production systems.

Keywords: Material flow simulation, optimization, machine learning, metaheuristics, metamodeling

1. Introduction

The utilization of analysis tools to support decision-making has become essential for companies to evaluate dynamic systems in a production environment. Thus, simulation is an established tool for analyzing complex relationships in production and logistics. Especially in the automotive industry, simulation is a commonly used tool due to the growing complexity in their production and logistics systems (Gutenschwager et al., 2017). For analyzing such systems, different parameters, constraints, and multiple objective functions are taken into

consideration. When the number of parameter combinations rises exponentially, evaluating all possible solutions becomes impractical. In these cases, it is beneficial to employ smart experiment design to effectively explore the parameter space. One concept for exploring large parameter spaces and gaining insights is Data Farming (Feldkamp et al., 2017). After reducing the number of necessary simulation experiments, simulation combined with optimization can be applied for refining and identifying optimal solutions (Barton, 2009; März et al., 2011). The objective of optimization is the determination of the best possible solution from a range of possibilities by using algorithmic methods (VDI, 2020). In particular, the combination of optimization and simulation is often used as a decision support system for solving industrial problems such as resource allocation or buffer allocation (Soares do Amaral et al., 2022; Yelkenci Kose & Kilincci, 2020).

However, existing simulation optimization methods, especially when integrating simulation into the optimization, are often time-consuming and inefficient for large and complex problems (Liu et al., 2018). This is because even hybrid-simulations still face the challenge of computational effort as optimization algorithms such as metaheuristics require many simulation runs for evaluation if they are combined with simulation directly (Sobottka et al., 2019). Therefore, the utilization of metamodeling approaches representing the input and output relations of particular production planning problems has received increasing attention in recent years. Besides, the need to improve the transferability and generalization of simulation optimization approaches to enhance their applicability in practice is pointed out.

As a result, the major research goal of this paper is to develop an approach that is time-efficient, transferable but still accurate for addressing different

optimization problems occurring in automotive production planning.

To this end, Section 2 discusses the background of previous work in the field of simulation optimization and metamodel-based optimization. Section 3 introduces the intended methodology to combine metamodeling and optimization and dives deeper into the algorithms for building metamodels. Furthermore, it discusses the algorithms used to optimize the production planning problems and their integration into the metamodel. Section 4 applies the presented approach to a use case in the automotive industry, and Section 5 focuses on the transferability of the approach. Section 6 presents the conclusion, and finally, Section 7 provides a brief outlook on further research opportunities.

2. Background and related work

2.1 Simulation and optimization

The goal of simulation optimization is to determine the input parameters of a complex simulation model that lead to a desired output (Medaglia et al., 2002). Thus, the integration of simulation into the optimization is referred to as simulation optimization (Hanschke & Zisgen, 2015). For implementing simulation optimization, there exist several strategies. The selection depends on the nature of the optimization model response function and the feasible region of the design parameters of the system being simulated (Amaran et al., 2016; Barton & Meckesheimer, 2006). For addressing discrete design parameters, appropriate strategies include ranking and selection, random search, metaheuristics, direct search and model-based methods. For the application of continuous design parameters, recommended strategies are gradient-based methods or metamodel-based methods. But also metaheuristics, direct search and model-based methods are applicable for continuous variables. Furthermore, an additional selection criterion refers to the intended solution finding, either global solution finding or local solution finding (Barton & Meckesheimer, 2006). Thereby, global methods such as mathematical optimization are computationally expensive as they seek for the global optimum which however can be impractical for implementation in real production systems.

In literature, the most popular and widely used method is the model-based approach (Soares do Amaral et al., 2022). The model-based approach implies that optimization and simulation modules are directly integrated, where optimization selects the solutions which need to be evaluated by simulation (Soares do Amaral et al., 2022). Within industrial

engineering problems, most common approaches are heuristics and metaheuristics (Trigueiro et al., 2020). Kassoul et al. (2022) apply genetic algorithms, tabu search and particle swarm algorithms to solve the buffer allocation problem in manufacturing lines. They aim to overcome the difficulty of increasing problem size and computational time. Süße et al. (2024) explore the application of genetic algorithms to improve a simulation of an automotive paint shop, in order to reduce energy consumption while at the same time ensuring on time delivery. Besides metaheuristics as single solver, the use of hybrid approaches has increased. These include hyper heuristics, the combination of multiple heuristics within the same optimization approach (Soares do Amaral et al., 2022), learning heuristics, that combine metaheuristics and machine learning (Liao et al., 2020) or individual application of machine learning algorithms for simulation optimization such as reinforcement learning (Rabe & Dross, 2015).

However, hybrid approaches still face the challenge of computational effort which remains a major limitation for simulation-based optimizations (Sobottka et al., 2019). This is because optimization algorithms such as metaheuristics require many simulation runs for evaluation when they are combined with simulation directly. Furthermore, the quality relies on the number of available evaluations. As a result, metamodeling approaches that serve as a surrogate simulation model have received increasing attention in the last years due to their potential to decrease computational time and therefore, enable fast decision-making.

2.2 Metamodel-based simulation optimization

Metamodels are simpler models that represent the input-output behavior of simulation models based on mathematical approximation (Barton, 2020). This enables the reduction of computational time that is spent on the optimization for the underlying problems. Different types of metamodels have been presented in the literature, such as linear regression metamodels, Gaussian process regression, stochastic kriging models or neural networks (Barton, 2009). The choice of metamodel type depends on the purpose of the metamodel and further influences the experiment design to fit the metamodel. Besides the approximation of simulation models, the results of the metamodels have the potential to further improve decision-making processes by conducting predictions, analysis and optimization.

In the area of manufacturing, the implementation of metamodels for analyzing dynamic systems is widely applied. Murphy et al (2019) apply

metamodeling for estimating order flowtimes in manufacturing systems based on discrete event simulation models. They execute 10.000 simulation runs to generate a dataset for training Artificial Neural Networks (ANN) Random Forest (RF) and Gradient Boosting algorithms. Bergmann et al. (2015) apply several machine learning algorithms, such as k-Nearest Neighbors (KNN), Support Vector Machines (SVM) and ANNs for the emulation of control strategies in manufacturing simulation models and conclude that these algorithms are capable of creating good approximations of the simulation model. Furthermore, Allmendinger & Pfaff (2018) implement gradient boosting algorithms and neural networks to build metamodels for an industry use case of a production system with four process parameters. Their research shows promising results for gradient boosting and neural networks as metamodels for approximation of production systems with a 99.7% accurate throughput prediction. A concrete application in the automotive industry is presented by Dengiz et al. (2016). They implement a metamodel approach for an automotive paint shop, where they apply full factorial design as design of experiments and polynomial regression metamodels. The optimization is performed with mathematical programming. Soares do Amaral et al. (2022) introduce a case study for a midsize factory of a car components company. They achieved the best results regarding the optimization objective with the random forest trained metamodel.

Despite these examples, studies that utilize machine learning for simulation optimization often put the focus on training a better predictive model rather than performing optimization (Amaran et al., 2016). In addition, the missing transferability of metamodels as surrogate models is pointed out by Rueden et al. (2020). They emphasize that metamodels are built for very specific purposes instead of exploiting the underlying system for similarities. Furthermore, the lack of practical implementation and deployment is mentioned (Trigueiro et al., 2020). Therefore, this paper focuses on investigating the application of metamodels for system optimization and the transferability of the approach to multiple problems in a practical environment. Thus, the application of optimization algorithms using the metamodel for system improvement is of particular interest.

3. Methodology

Based on Soares do Amaral et al. (2022) and Barton & Meckesheimer (2006), the methodology as shown in Figure 1 was developed to implement the metamodel-based simulation optimization for the industry use cases. After building a **simulation model**

for representing the intended planning problem and its system dynamics (Step 1), a **metamodeling type** is identified (Step 2). Most commonly applied metamodel types in the literature of simulation optimization are Kriging models, followed by Polynomial Regression and ANN. Though Kriging and Polynomial Regression seem to perform worse on large and more complex problem types (Soares do Amaral et al., 2022).

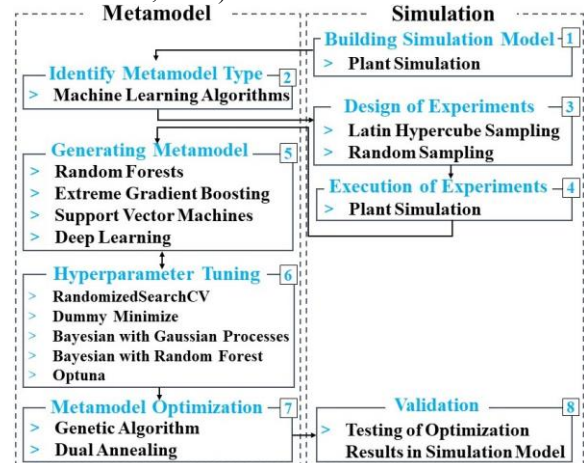


Figure 1: Methodology for implementing metamodel-based simulation optimization.

Therefore, and due to previous successful implementations in the manufacturing sector, this paper makes use of machine learning algorithms to build the surrogate models. In addition, they provide the possibility to include further analysis opportunities such as data farming (Feldkamp et al., 2017).

Before starting the experiments, the **design of experiments** (DOE) ensures that the entire search space is represented without executing large amounts of experiments (Step 3). Especially for aiming at the development of a practical approach, time reduction is a major factor. There exist several methods regarding the DOE such as factorial design, random sampling (RS) or Latin Hypercube Sampling (LHS) (Montgomery, 2017). LHS generates high orthogonality and has good space filling properties, which is necessary to mitigate correlations between two input parameters (McKay et al., 1979). With regard to previous studies in the field of metamodeling, LHS, but also RS have proven successful implementations for addressing manufacturing problems. Besides, steps 1, 3, and 4 are all subject to verification and validation during the development. They are however not considered in detail in the context of this article, since the actual validation step relates to the testing of the optimization results.

By using the results of the simulation runs as training dataset, the **generation of the metamodels**

follows. Due to the previously introduced studies (Allmendinger & Pfaff, 2018; Soares do Amaral et al., 2022), Random Forest Regressors (RFR), Extreme Gradient Boosting (XGB), Support Vector Regressors (SVR) and Deep Neural Networks (DNN) are applied for creating the regression models that predict the output data. RFR use multiple decision trees and have proven their efficiency to handle large datasets with higher dimensionality (Breiman, 2001). The ability of XGB to deliver high predictive accuracy and its versatility in handling a variety of data types and structures makes it a suitable algorithm to test the application on multiple problems (Chen & Guestrin, 2016). SVRs are effective in high-dimensional spaces and provide a robust algorithm, which is able to handle various data types (Schölkopf & Smola, 2009). Neural networks are computational models inspired by the human brain, composed of interconnected nodes that process and transmit information through weighted connections. As Soares do Amaral et al. (2022) mentioned, DNNs are an interesting open research area, so the application of DNN is tested. One of the main advantages of DNNs is their ability to perform feature engineering and selection automatically from raw data (Owen, 2022), which can be useful for general deployment approaches.

Hyperparameter tuning is a critical process in machine learning aimed at improving model performance by systematically tuning hyperparameters. Unlike model parameters, which are learned during training, hyperparameters are set prior to training and control the learning process, such as the learning rate or regularization strength (Owen, 2022). Effective hyperparameter optimization techniques include grid search, random search, and more advanced methods like Bayesian optimization and genetic algorithms. Optimizing hyperparameters enhances model accuracy, generalization, and efficiency, resulting in a more robust model. Soares do Amaral et al. (2022) point out that hyperparameter tuning improved the performance of their machine learning-based metamodels and the selection of hyperparameter ranges is based on their results.

Optimization algorithms serve to find the optimal configurations of parameters to reach a desired boundary condition while minimizing or maximizing a specific objective function. Previous studies in the field of simulation optimization using metamodels often applied genetic algorithms (Soares do Amaral et al., 2022). They are a class of optimization techniques inspired by the principles of natural selection and genetics, effectively searching for optimal or near-optimal solutions in complex problem spaces (Goldberg & Holland, 1988). Alternatively, simulated annealing, a probabilistic optimization technique

inspired by the annealing process in metallurgy, is applied. It explores the solution space by allowing occasional uphill moves, enabling it to escape local optima, with the probability of such moves decreasing over time to converge on an optimal or near-optimal solution (Kirkpatrick et al., 1983). In particular dual annealing, which combines the traditional simulated annealing algorithm with local search heuristics, enhances convergence and solution quality. The balancing of global exploration with local exploitation leads to even more accurate and efficient optimization results.

Finally, the initial simulation study serves to **validate** the results of the optimization calculated with the metamodels. Therefore, the parameter combinations suggested by the optimization algorithm are entered into the simulation. In this way, we can elaborate on the one hand, if the simulation reaches the demanded throughput with the proposed solution, and on the other hand the difference in predicted throughput provides an indication about prediction accuracy.

4. Production planning problems in the automotive industry

To test the applicability of the presented approach, two use cases from an automotive company are introduced: an assembly shop and a body shop.

Both shops consist of production lines operating at specified speeds, which are subject to stochastic machine failures. Buffers between the lines provide decoupling to minimize the impact of line failures on the shop's overall throughput. Throughput, measured as the number of vehicles produced per hour, is the key performance target for the entire production. The goal is to meet this production target robustly at the lowest possible cost. Automotive production involves a variety of cost factors. These use cases focus on infrastructure costs related to buffer sizing and operational costs associated with cycle times. Larger buffers result in higher costs, which emphasizes the need to minimize buffers sizes. Conversely, faster cycle times result in higher capacity costs.

Therefore, the optimization objective is to minimize the costs associated with buffer sizing and cycle times while maintaining the specified throughput. The inputs for the optimization include line cycle times, possible buffer capacities between the lines, and the desired throughput. The decision variables are the buffer capacities and cycle time variations. The outputs are the optimal buffer capacities, cycle times and the achieved throughput.

Given the large number of parameters and stochastic elements, the dynamic behavior inside the shop is highly complex. As analytical models come to

their limits for these problems, a discrete event simulation is employed to represent the shops.

The first use case serves to illustrate the feasibility of the approach, whereas the second use case presents a first step towards testing its transferability.

4.1 Assembly shop

The assembly shop consists of seven main lines, each containing several workstations. Buffers with varying capacities are positioned between the lines. To represent cycle times, the parameter of line capacity reserve is examined. This parameter directly correlates with the availability of the lines and their decoupling. It reflects the ability to speed up the cycle time for particular lines. To compensate for the losses caused by line failures, line capacity must be increased. It ensures a robust throughput despite breakdowns by increasing capacity (Daferner et al., 2021). In this use case, the line capacity reserve ranges from 0.010 and 0.060, indicating a cycle time acceleration of 1% to 6% for the respective lines. Buffer configurations range from one to eight. The objective function evaluates proposed configurations based on a cost function (C). For this study, the cost (C) is calculated using the cost of the buffer (c_b), the total number of buffers (B_n), cost of line capacity increase (c_v) and the total line capacity increases (v_l) as shown in (1).

$$C_1 = c_b \sum B_n + c_v \sum v_l \quad (1)$$

When defining c_b and c_v , amortization needs to be considered. If the throughput target is not achieved with the proposed parameter configuration, an additional penalty will be added to the cost. It is defined as the highest cost value, which implies all maximum buffer capacities and the maximum line capacity increase, as this leads to a minimum cycle time. The values for costs are fictional and serve for the comparison among the algorithms.

4.2 Design of experiments

To minimize the number of experiments while still achieving comprehensive coverage of the parameter space, LHS and RS were employed. Considering the range of possibilities for combining the individual input parameters and time constraints for the practical approach, 10'000 simulation runs are created with three replications for each run, resulting in 30'000 runs. Within the simulation, the model simulation runs for 81 days and the models' settling time is 21 days. The runs were executed with Tecnomatix Plant Simulation 23.

4.3 Metamodels and hyperparameter tuning

Metamodels are subsequently generated using machine learning algorithms based on the simulated training data. They represent regression models that serve to predict the throughput of the assembly shop. The implementation of the machine learning algorithms with Python 3.10 and the scientific libraries Scikit-Learn, Tensorflow and Keras was an interrelated process with hyperparameter optimization to tune the model's performance. Six hyperparameter optimization methods were tested on the training data: Default values, Randomized Search Cross Validation (RSCV), Dummy minimize, Bayesian optimization with Gaussian processes (GP) and random forest (RF) and lastly, Optuna (Rimal et al., 2024). For evaluating the performance of the models, the Mean Absolute Error (MAE) was applied. The MAE (2) serves as performance metric for the machine learning models. It is a widely used metric for regression models that measures the average magnitude of the errors between predicted values (y_1) and actual values (\hat{y}_i), without considering their direction. It averages the absolute differences, which makes it robust to outliers. For calculating, n refers to the number of errors.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_1 - \hat{y}_i| \quad (2)$$

The influence of individual hyperparameters on the model is illustrated in Figure 2.

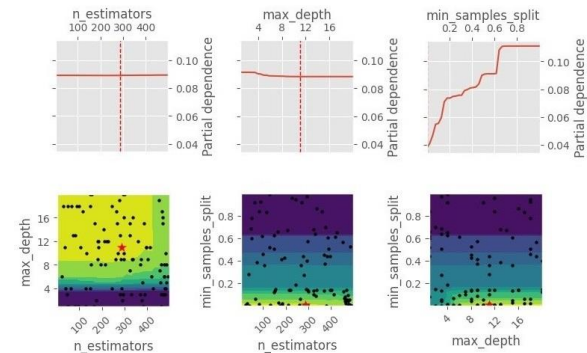


Figure 2: Influence of random forest hyperparameters using Bayesian optimization with random forests.

The plot on the top left shows the partial dependence of MAE on the number of estimators. The relatively flat line indicates that this parameter does not have a significant effect on the model performance. The same is true for the maximum depth. On the other hand, the model performance seems to improve by decreasing the number of minimum samples split. The contour plots illustrate the relations

between hyperparameter combinations, where the color represents the magnitude of partial dependence with darker colors indicating higher MAE values. The optimal combination of hyperparameters is marked with a red star. This combination minimizes the performance metric MAE of the Random Forest model. Figure 3 presents the MAEs for training and testing data for the applied hyperparameter tuning methods on the Random Forest model. Comparing the values of MAE for the training and testing data, it becomes visible that using the default parameters leads to an overfitted model, as well as for Optuna.

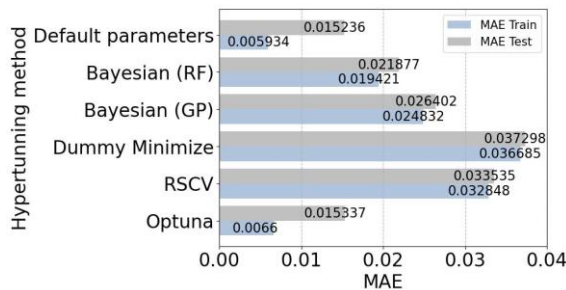


Figure 3: Hyperparameter optimization on Random Forest model.

As the MAE train is much smaller than the MAE test with the default values, it indicates that the training model is not generalized well enough, resulting in a model that memorizes instead of learning the relations between input and output variables. Bayesian optimization with random forests achieved the best parameter configuration with the lowest MAE without overfitting and demonstrates better robustness for future predictions. After performing hyperparameter tuning for all implemented machine learning algorithms, the configurations of parameters leading to the lowest MAE without overfitting were set as model parameters. This ensures more accurate results for the final optimization.

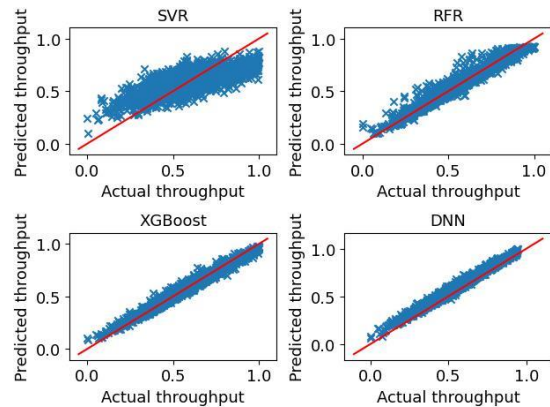


Figure 4: Comparison of predictions of trained machine learning algorithms with simulation results on test data for LHS.

Based on this, the blue data points in Figure 4 present the prediction results of the machine learning algorithms on the test dataset, compared to the actual results with normalized distribution. Data points on the red line indicate an optimal prediction without deviation. XGB predicted the output values most accurately, followed by DNN and RFR. The prediction using SVR was less accurate. However, the MAE is low for all models as shown in Table 1.

Table 1: MAEs on train and test data of the implemented machine learning algorithms.

	Sampling	SVR	RFR	XGB	DNN
Train	LHS	0.0735	0.0194	0.0141	0.0181
	RS	0.0710	0.0243	0.0140	0.0162
Test	LHS	0.0765	0.0219	0.0161	0.0196
	RS	0.0774	0.0269	0.0162	0.0162

The table presents the concrete values for MAE test and MAE train for LHS and RS. The MAE test for XGB indicates that the prediction values deviate by 0.016 jobs per hour (JPH) from the actual simulation result in the test dataset. Furthermore, the difference in MAEs for using LHS or RS is marginal, concluding that both sampling methods seem appropriate for the use case.

4.4 Optimization and validation

As previously mentioned, the objective of the case study is to find the optimal configuration of buffers and line capacity increase to meet the throughput at the lowest cost.

The results of the optimization algorithms varied, depending on the approach and metamodel. As shown in Table 2, using a genetic algorithm for optimization, DNN provided buffer and cycle time configurations with the lowest cost, followed by RFR, XGB and SVR. Dual Annealing achieved similar results.

Table 2: Comparison of costs generated with dual annealing and genetic algorithm.

	SVR	RFR	XGB	DNN
GA	64.32	47.84	56.24	13.05
Dual annealing	84.45	47.40	61.21	10.50

For the validation, the proposed configurations were used as input parameters in the original simulation model. Regarding the difference between predicted throughput and actual throughput, DNN had the greatest deviation with 0.24 JPH. The prediction of the throughput by RF was also lower than the simulation result, with a difference of 0.13 JPH. For both of these solutions, the target was not achieved. Therefore, it might be useful to increase the target

throughput of the optimization to ensure that the target is achieved, considering the small error of the metamodels. However, SVR and XGB both predicted results meeting the target throughput. SVR had a deviation of 0.16 JPH and XGB achieved the best performance with a deviation of 0.08 JPH above the target and providing a solution with lower cost.

In terms of computational efficiency, the optimization using the metamodels was significantly faster. Considering the XGB metamodel, the entire calculation, including training with 1000 iterations and hyperparameter optimization using RSCV, took 138.8s. The genetic algorithm ran for 150 generations with a population size of 100. In comparison, the same optimization integrated directly into the simulation took 461 minutes. Additionally, the costs were higher compared to the metamodels as shown in Figure 5.

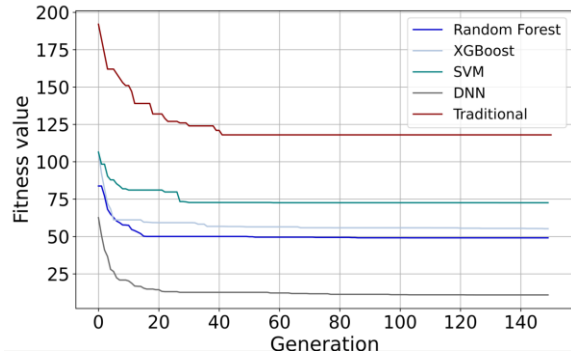


Figure 5: Development of fitness value of the genetic algorithm.

It should be noted, however, that generating the training data for the machine learning model required an additional one-time computational effort of 309 minutes. Consequently, if the simulation runtime is relatively short, it may be more efficient to perform the optimization directly within the simulation, particularly for one-time optimization tasks. Additionally, the runtime of the optimization within the simulation can be reduced by decreasing the number of generations in the GA. However, this resulted in higher associated costs.

5. Transferability of the approach

In order to test the possibility to transfer the proposed approach, a second use case is introduced. It represents a body shop, which consists of a highly automated area equipped with robots. This area includes subareas with several workstations. Additionally, the area contains multiple buffers.

Similar to the assembly shop, the aim is to achieve the desired throughput while minimizing the cost for the buffer configurations and the cycle times.

However, the underlying system dynamics influencing the parameters vary. The throughput serves as the result parameter. The adjustable input process parameters consist of seven buffers (B_1 to B_7) with different capacities and three cycle times (t_1 to t_3) for the subareas. These cycle times can take on five different values.

Besides, in comparison to the assembly line an additional constraint for the optimization is tested. For some buffers, the cost increases once a certain number is reached (β), as the increasing of buffers over that number implies additional investment costs (c_i). The following cost function for optimizing buffer capacities and cycle times, including cost of cycle times (c_t) and cost of buffers (c_b) is defined as

$$C_2 = c_b \sum B_n + c_t \sum t_n + \sum \Theta(B_n - \beta_n) c_{i,n} \quad (3)$$

Thereby, the calculation of the sum of additional investment costs Θ is subject to the Heaviside function

$$\Theta(x) = \begin{cases} 0 & : x < 0 \\ 1 & : x \geq 0 \end{cases} \quad (4)$$

The experimental design and execution of simulation runs is as presented in 4.2.

5.1 Body shop results

For generating and training the metamodels, the steps as described in section 4 are executed. Regarding the performance of the metamodels for the body shop, XGB provided the smallest MAE, followed by DNN, SVR and RF having the highest MAE as shown in Table 3. As with the previous case study, the sampling methods do not seem to have a great influence on the final model performance.

Table 3: MAEs on train and test data of the implemented machine learning algorithms.

	Sampling	SVR	RFR	XGB	DNN
Train	LHS	0.0332	0.0401	0.0240	0.0293
	RS	0.0336	0.0402	0.0240	0.0511
Test	LHS	0.0330	0.0566	0.0262	0.0292
	RS	0.0344	0.0560	0.0275	0.0511

For the optimization, the objective is to minimize cost of buffers and cycle times, while maintaining the desired throughput and meeting the defined constraints. The development of the fitness value, which measures the cost of each proposed solution candidate, shows Figure 6. The genetic algorithm using the metamodel generated with the DNN provides the solution with lowest cost, followed by RF, XGB and SVR proposing a solution with the highest cost.

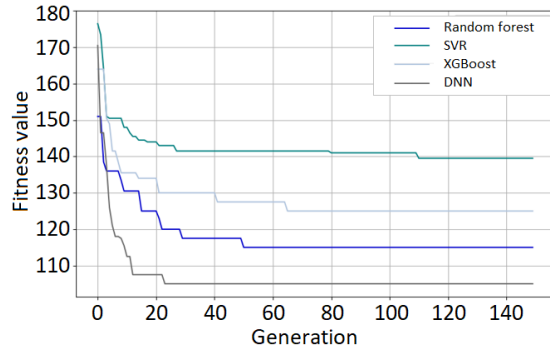


Figure 6: Comparison of fitness value development of the genetic algorithm.

Testing the proposed parameter configurations in the simulation shows that the solutions provided by the metamodels generated with DNN, RF and XGB do not reach the target value. DNN had the greatest deviation with 0.62 JPH, RF deviated by 0.41 JPH and XGB by 0.18 JPH. Furthermore, even if the proposed candidate solution of SVR provided a result achieving the target, it still deviated by 0.5 JPH from the proposed target value. The deviation might occur due to the noise within the data created by simulations when only executing a limited number of evaluations per simulation run. This has effects on the training data used for generating the metamodels as it can be sensitive to outliers.

An approach for making the metamodels more robust without increasing simulation runs is to use quantile regression (Batur et al., 2018). Quantile regression (QR) estimates the conditional median or other quantiles of the dependent variable. In this way, the entire variability and distribution of the target variable across different quantiles can be examined. For testing the approach, we integrated QR into the generation of an extreme gradient boosted tree model. The application of the genetic algorithm to this model provided a solution that only deviated by 0.02 JPH when entered in the initial simulation model and achieved the required throughput. This is significantly lower compared to the metamodels trained without QR, as shown in Figure 7. Applying dual annealing

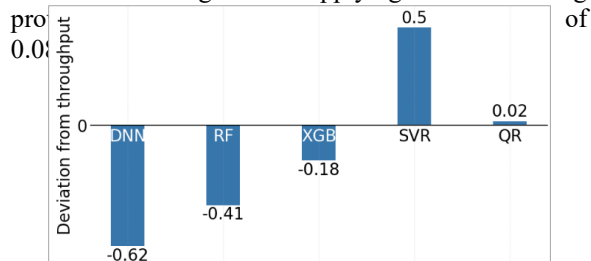


Figure 7: Deviation from predicted throughput using metamodels with and without QR.

Furthermore, the cost of the provided solution using dual annealing was 17% lower compared to the solution candidate of the genetic algorithm. The solution provided by dual annealing achieved in comparison to an optimization directly integrated into the simulation a buffer and cycle time configuration with 11.5% lower costs, and the runtime was 40 times shorter. The decrease in computational time while at the same time providing a more cost-efficient solution is promising for repeating optimization tasks.

5.2 Practical implementation

When considering future implementations in the industry, it is important to evaluate the maturity of a shop layout before opting for a metamodel-based optimization approach. Given the long execution times required for simulation experiments to generate the initial training data for the metamodel, it is recommended to apply metamodels at a later stage in the planning process. This helps avoid repeating the data generation process, which can take several hours. Once the layout is finalized and the focus shifts to optimization, the metamodeling approach becomes beneficial due to its fast computation and cost-efficient results.

6. Conclusion

By comparing the metamodels for both studied problems, it became clear that the performance of individual algorithms depends on the dataset and problem type. Therefore, it is difficult to decide for one algorithm in advance. It is rather recommended to train the dataset on multiple algorithms and decide for the algorithm achieving the lowest MAE. An algorithm selection process can support automating this step, because once the algorithms are set up and include hyperparameter optimization, there is no additional input of the user necessary.

With regard to the model set up, DNN turned out to be the most challenging algorithm due to its sensitivity for parameters. In addition, it is difficult in terms of reproducibility. Therefore, DNNs might be demanding for practical implementation, as it requires domain knowledge for correct adjustment. Hyperparameter optimization improved the performance of the algorithms, and especially reduced overfitting of the models, confirming the results of Soares do Amaral et al. (2022).

Regarding the overall results of the optimization, using the XGB metamodel was most reliable when predicting optimization configurations. DNN and RF were too optimistic when predicting configurations for the input parameters. Therefore, it is recommended to

either increase the target value for the optimization, or to investigate outliers at the beginning of the study with quantile regression. In this way, the noise of simulation results could be reduced, resulting in increased optimization accuracy.

One of the main benefits of the entire approach is the fast execution of the optimization. Compared to an optimization directly integrated in the simulation, run time can be reduced significantly from hours to seconds when analyzing existing layouts with already trained metamodels. This supports the rapid decision-making in the fast-changing industry environment. Furthermore, the approach allows changes in the optimization parameters, for instance the cost function, and is able to provide optimization proposals without the need of additional simulation runs. In addition, it can be implemented in such a way that users can apply it without having deep domain expertise, supporting widespread use for industry application.

On the other side, as soon as there are changes in the entire system, for instance adaptations to the availabilities or additional workstations, the meta-model needs to be retrained with new datasets of the adjusted simulation. Therefore, it is recommended to use this approach at a later planning stage and especially for large and complex simulation models.

7. Outlook and future research

For future studies, the implementation of logistics use cases is of great interest. Logistic problems consist of a variety of parameters and their influence on the entire approach provides investigation potential. As the large number of parameters in logistic simulations results in slow simulation models, the presented approach provides great potential to reduce time invested in optimization. Also, the exploration of multiple objectives within the approach is an interesting research field.

Furthermore, the integration of data farming concepts into the framework can enhance its practical impact even more. Besides, a detailed exploration of the effects of cost functions for the final optimization of the production planning problem provides potential for future research. The impact on the objective function, fitness and penalty functions depending on different problem types can be examined. Furthermore, the influence of robustness is another research area to explore more in detail to reduce the error in the final optimization procedure.

8. References

Allmendinger, B., & Pfaff, S. Approximation von Materialflussmodellen durch neuronale Netze und

- Gradient-Boost Verfahren. In *Tagungsband ASIM Workshop STS/GMMS 2018* (pp. 191–196). (Original work published 2018)
- Amaran, S., Sahinidis, N. V., Sharda, B., & Bury, S. J. (2016). Simulation optimization: a review of algorithms and applications. *Annals of Operations Research*, 240(1), 351–380. <https://doi.org/10.1007/s10479-015-2019-x>
- Barton, R. R. (2009). Simulation optimization using metamodels. *Proceedings of the 2009 Winter Simulation Conference (WSC)*, 230–238.
- Barton, R. R. (2020). Tutorial: Metamodeling for Simulation. In *2020 Winter Simulation Conference (WSC)*.
- Barton, R. R., & Meckesheimer, M. (2006). Chapter 18 Metamodel-Based Simulation Optimization. In S. G. Henderson & B. L. Nelson (Eds.), *Handbooks in Operations Research and Management Science : Simulation* (Vol. 13, pp. 535–574). Elsevier. [https://doi.org/10.1016/S0927-0507\(06\)13018-2](https://doi.org/10.1016/S0927-0507(06)13018-2)
- Batur, D., Bekki, J. M., & Chen, X. (2018). Quantile regression metamodeling: Toward improved responsiveness in the high-tech electronics manufacturing industry. *European Journal of Operational Research*, 264(1), 212–224. <https://doi.org/10.1016/j.ejor.2017.06.020>
- Bergmann, S., Feldkamp, N., Straßburger, S., & Hinze, U. (2015). *Abbildung von Steuerungslogiken durch maschinelles Lernen für die Simulation von Produktionssystemen (Emulation of Control Strategies through Machine Learning in Manufacturing Simulations)*.
- Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- Daferner, M., Mesenhöller, E., & Schmidt, T. (2021). Strukturierung eines Montagesystems bei Fertigungszeitspreizung. *Teil: Fertigungsflexibilität Als Wettbewerbsvorteil Im Fließbetrieb*, 116(4), 237–241. <https://doi.org/10.1515/zwf-2021-0051>
- Dengiz, B., İc, Y. T., & Belgin, O. (2016). A meta-model based simulation optimization using hybrid simulation-analytical modeling to increase the productivity in automotive industry. *Mathematics and Computers in Simulation*, 120, 120–128. <https://doi.org/10.1016/j.matcom.2015.07.005>
- Feldkamp, N., Bergmann, S., Schulze, T., & Straßburger, S. (2017). *Data Farming im Kontext von Produktion und Logistik*.
- Goldberg, D. E., & Holland, J. H. (1988). Genetic Algorithms and Machine Learning. *Machine Learning*, 3(2), 95–99. <https://doi.org/10.1023/A:1022602019183>
- Gutenschwager, K., Rabe, M., Spieckermann, S., & Wenzel, S. (2017). *Simulation in Produktion und Logistik*:

- Grundlagen und Anwendungen.*
<https://doi.org/10.1007/978-3-662-55745-7>
- Hanschke, T., & Zisgen, H. Verknüpfung von Simulation und Optimierung: Kategorien und Beispiele - ein Bericht über die VDI Richtlinie 3633 Blatt 12. In *Simulation in Production and Logistics 2015* (pp. 111–118). (Original work published 2015)
- Kassoul, K., Cheikhrouhou, N., & Zufferey, N. (2022). Buffer allocation design for unreliable production lines using genetic algorithm and finite perturbation analysis. *International Journal of Production Research*, *60*(10), 3001–3017. <https://doi.org/10.1080/00207543.2021.1909169>
- Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science (New York, N.Y.)*, *220*(4598), 671–680. <https://doi.org/10.1126/science.220.4598.671>
- Liao, X., Shi, J., Li, Z., Zhang, L., & Xia, B. (2020). A Model-Driven Deep Reinforcement Learning Heuristic Algorithm for Resource Allocation in Ultra-Dense Cellular Networks. *IEEE Transactions on Vehicular Technology*, *69*(1), 983–997. <https://doi.org/10.1109/TVT.2019.2954538>
- Liu, R., Xie, X., Yu, K., & Hu, Q. (2018). A survey on simulation optimization for the manufacturing system operation. *International Journal of Modelling and Simulation*, *38*(2), 116–127. <https://doi.org/10.1080/02286203.2017.1401418>
- März, L., Krug, W., Rose, O., & Weigert, G. (2011). Simulation und Optimierung in Produktion und Logistik. *Simulation Und Optimierung in Produktion Und Logistik: Praxisorientierter Leitfaden Mit Fallbeispielen, VDI-Buch, ISBN 978-3-642-14535-3. Springer-Verlag Berlin Heidelberg, 2011, -I.* <https://doi.org/10.1007/978-3-642-14536-0>
- McKay, M. D., Beckman, R. J., & Conover, W. J. (1979). A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, *21*(2), 239–245. <https://doi.org/10.2307/1268522>
- Medaglia, A. L., Fang, S.-C., & Nuttle, H. L.W. (2002). Fuzzy controlled simulation optimization. *Fuzzy Sets and Systems*, *127*(1), 65–84. [https://doi.org/10.1016/S0165-0114\(01\)00153-1](https://doi.org/10.1016/S0165-0114(01)00153-1)
- Montgomery, D. C. (2017). *Design and analysis of experiments*. John Wiley & sons.
- Murphy, R., Newell, A., Hargaden, V., & Papakostas, N. (2019). Machine learning technologies for order flowtime estimation in manufacturing systems. *Procedia CIRP*, *81*, 701–706. <https://doi.org/10.1016/j.procir.2019.03.179>
- Owen, L. (2022). *Hyperparameter Tuning with Python: Boost your machine learning model's performance via hyperparameter tuning*. Packt Publishing. <https://books.google.de/books?id=CqF-EAAAQBAJ>
- Rabe, M., & Dross, F. (2015). A Reinforcement Learning approach for a Decision Support System for logistics networks. In *2015 Winter Simulation Conference (WSC)*.
- Rimal, Y., Sharma, N., & Alsadoon, A. (2024). The accuracy of machine learning models relies on hyperparameter tuning. *Multimedia Tools and Applications*. Advance online publication. <https://doi.org/10.1007/s11042-024-18426-2>
- Rueden, L. von, Mayer, S., Sifa, R., Bauckhage, C., & Garcke, J. (2020). Combining Machine Learning and Simulation to a Hybrid Modelling Approach: Current and Future Directions. In M. R. Berthold, A. Feelders, & G. Krempf (Eds.), *Advances in Intelligent Data Analysis XVIII* (pp. 548–560). Springer International Publishing.
- Schölkopf, B., & Smola, A. J. (2009). *Learning with kernels: Support vector machines, regularization, optimization and beyond. Adaptive computation and machine learning*. MIT Press.
- Soares do Amaral, João Victor, Carvalho Miranda, R. de, Montevechi, J. A. B., dos Santos, C. H., & Gabriel, G. T. (2022). Metamodeling-based simulation optimization in manufacturing problems: a comparative study. *The International Journal of Advanced Manufacturing Technology*, *120*(7), 5205–5224. <https://doi.org/10.1007/s00170-022-09072-9>
- Soares do Amaral, João Victor, Montevechi, J. A. B., Miranda, R. d. C., & Junior, Wilson Trigueiro de Sousa (2022). Metamodel-based simulation optimization: A systematic literature review. *Simulation Modelling Practice and Theory*, *114*, 102403. <https://doi.org/10.1016/j.simpat.2021.102403>
- Sobottka, T., Kamhuber, F., Faezirad, M., & Sihm, W. (2019). Potential for Machine Learning in Optimized Production Planning with Hybrid Simulation. *Procedia Manufacturing*, *39*, 1844–1853. <https://doi.org/10.1016/j.promfg.2020.01.254>
- Süße, M., Xie, X., & Ihlenfeldt, S. (2024). Combining material flow simulation and optimization for sustainable manufacturing – application in automotive paint shops. *Procedia CIRP*, *122*, 659–664. <https://doi.org/10.1016/j.procir.2024.01.092>
- Trigueiro, W., Montevechi, J. A. B., Miranda, R. d. C., Oliveira, Mona Liza Moura de, & Campos, A. T. (2020). Shop floor simulation optimization using machine learning to improve parallel metaheuristics. *Expert Systems with Applications*, *150*, 113272. <https://doi.org/10.1016/j.eswa.2020.113272>
- VDI. (2020). *Simulation of systems in materials handling, logistics and production - Simulation and optimisation: VDI 3633 Blatt 12*.
- Yelkenci Kose, S., & Kilincci, O. (2020). A multi-objective hybrid evolutionary approach for buffer allocation in open serial production lines. *Journal of Intelligent Manufacturing*, *31*(1), 33–51. <https://doi.org/10.1007/s10845-018-1435-6>