Surrogate-based automated hyperparameter optimization for expensive automotive crashworthiness optimization

Long, F.X.; Stein, N. van; Frenzel, M.; Krause, P.; Gitterle, M.; Bäck, T.H.W.

Citation
Long, F. X., Stein, N. van, Frenzel, M., Krause, P., Gitterle, M., & Bäck, T. H. W. (2025). Surrogate-based automated hyperparameter optimization for expensive automotive crashworthiness optimization. *Structural And Multidisciplinary Optimization, 68*. doi:10.1007/s00158-025-03989-x

Version:	Publisher's Version
License:	Creative Commons CC BY 4.0 license
Downloaded from:	https://hdl.handle.net/1887/4292289

Note: To cite this publication please use the final published version (if applicable).

Structural and Multidisciplinary Optimization (2025) 68:68
https://doi.org/10.1007/s00158-025-03989-x

RESEARCH PAPER

Surrogate-based automated hyperparameter optimization for expensive automotive crashworthiness optimization

Fu Xing Long[1,2] · Niki van Stein[2] · Moritz Frenzel[3] · Peter Krause[4] · Markus Gitterle[5] · Thomas Bäck[2]

Received: 27 May 2024 / Revised: 14 February 2025 / Accepted: 3 March 2025 / Published online: 18 April 2025
© The Author(s) 2025

Abstract

In the automotive industry, solving crashworthiness optimization problems efficiently is crucial to minimize time and cost investment on expensive function evaluations, e.g., using simulation runs. Nonetheless, automotive crashworthiness optimization is time-consuming and challenging even with domain knowledge, due to the fact that crash problems are typically high-dimensional, nonlinear, and discontinuous. In this work, we propose an automated hyperparameter optimization (HPO) approach for expensive black-box optimization (BBO) problems that can assist practitioners to solve automotive crash problems efficiently using optimally configured optimization algorithms. Precisely, the landscape characteristics of BBO problems, e.g., quantified using exploratory landscape analysis (ELA), are analyzed to identify cheap-to-evaluate representative functions that belong to the same optimization problem class. Based on these representative functions, algorithm configurations can be optimally fine-tuned at a relatively low computational cost. Using three optimization algorithms, consisting of modular covariance matrix adaptation evolutionary strategy (CMA-ES), modular differential evolution (DE), and Bayesian optimization (BO), we evaluate the potential of our approach based on the black-box optimization benchmarking (BBOB) suite and an automotive side crash problem. Since the optimal configurations identified using our approach can perform well on most of the BBOB functions, we believe that our approach can generalize well to BBO problems with similar optimization complexity. For the automotive side crash problem, the BO configuration fine-tuned using our approach can outperform the default BO configuration as well as the conventional response surface method (RSM), in terms of the best-found-solution and convergence speed. Furthermore, better solutions can be identified using the proposed approach compared to successive RSM (SRSM), when dealing with complex crash functions and a limited function evaluation budget. With appropriate extensions, we are confident that our approach can be applied to other real-world expensive BBO domains beyond automotive crashworthiness optimization.

Keywords Black-box optimization · Hyperparameter optimization · Representative functions · Bayesian optimization · Exploratory landscape analysis · Automotive crashworthiness

1 Introduction

Generally, black-box optimization (BBO) belongs to a group of optimization problems, where an analytic form is not available, derivative information is lacking, and numerical approximation of the derivatives is costly (Audet and Hare 2017). In real-world engineering domains, such as crashworthiness optimization (Long et al. 2022) and control system calibration (Thomaser et al. 2022) in the automotive industry, BBO problems oftentimes require expensive function evaluations. Solving automotive crashworthiness optimization problems is often tremendously challenging and tedious, due to the fact that crash problems are strongly nonlinear, discontinuous,

Responsible editor: Palaniappan Ramu.

✉ Fu Xing Long
fu-xing.long@bmw.de

[1] BMW Group, Knorrstraße 147, Munich D-80788, Germany

[2] LIACS, Leiden University, Niels Bohrweg 1, NL-2333 Leiden, Netherlands

[3] Altair Engineering GmbH, Calwer Straße 7, D-71034 Böblingen, Germany

[4] divis intelligent solutions GmbH, Joseph-von-Fraunhofer-Straße 20, D-44227 Dortmund, Germany

[5] Munich University of Applied Sciences, Dachauer Straße 98b, D-80335 Munich, Germany

and high-dimensional as well as the function evaluations are costly and/or time-consuming, e.g., requiring finite element (FE) simulation runs. For instance, an automotive crash problem can be typically defined as minimization of vehicle weight, and thus, manufacturing costs, while fulfilling the increasingly stricter regulations on road safety, e.g., passengers must be sufficiently protected in the event of a crash (Duddeck 2008). On top of that, vehicle design for battery electric vehicle (BEV) is more sophisticated, where vehicle battery must be additionally protected from serious damage during impact due to safety reasons.

Classically, automotive crash problems are solved using a one-shot optimization approach, where simply the best sample of a design of experiments (DoE) is considered as the optimization solution (Bossek et al. 2020). The main advantage of this approach is that multiple DoE samples can be evaluated in parallel using FE simulation runs. Later on, the so-called response surface method (RSM) has gained popularity in the automotive industry, where response surfaces, such as polynomial function, radial basis function (RBF), and Gaussian process (GP) models, are constructed based on some DoE samples to approximate the true crash problems (Fang et al. 2017). Following this, solutions better than the best DoE sample or the global optimum can be predicted using these models. Despite substantial work has been invested (Stander et al. 2004; Fang et al. 2005; Pan and Zhu 2011), RSM is still ineffective for complex crash problems due to poor model fitting quality.

Based on RSM, the successive response surface method (SRSM) was introduced to sequentially re-sample within a subspace of the design space (Kok and Stander 1999). Principally, a subspace is defined within the design space, which can successively shrink and move toward regions with better solutions. In each iteration, a new response surface is constructed to approximate the true function within the subspace, similar to RSM. While SRSM has demonstrated potential in solving automotive crash problems (Kurtaran et al. 2002), its application in practice is still rather limited, e.g., due to the poorly fitted response surfaces, expensive computational cost using iterative re-sampling, and difficulty to transfer information between iterations (Fang et al. 2017).

On the other hand, multi-fidelity optimization (Kaps et al. 2022) and model order reduction (Czech et al. 2022) are other active research directions for crashworthiness optimization. Basically, the aim is to solve an optimization problem at a reduced cost by utilizing low-fidelity models that have a good balance between modeling accuracy and computational resources. Nonetheless, an application of these techniques for complex automotive crash problems can be particularly limited, e.g., due to low model accuracy similar to RSM. Subsequently, we focus on exploring alternatives that can effectively solve BBO problems from the perspective of fine-tuning algorithm configuration.

While population-based evolutionary algorithms (EA) are powerful in solving BBO problems, e.g., covariance matrix adaptation evolutionary strategy (CMA-ES) (Hansen and Ostermeier 1996) and differential evolution (DE) (Storn and Price 1997), a large function evaluation budget is usually required for optimization convergence. Consequently, they are less practical for expensive automotive crash problems (Yildiz and Solanki 2012). In the meanwhile, progressive developments have been attempted to overcome the limitation of EA in expensive BBO domains (Li et al. 2022).

On the other hand, Bayesian optimization (BO) (Mockus 1982), also known as efficient global optimization (EGO) (Jones et al. 1998), has shown promising potential in solving automotive crash problems (Hamza and Shalaby 2014; Sun et al. 2020). In brief, BO is an iterative optimization algorithm specifically designed for expensive BBO problems with a limited evaluation budget, which explores and searches for better solutions through re-sampling promising regions within the search space. Over the years, different variants of BO have been introduced, such as mixed-integer parallel EGO (MiP-EGO) (van Stein et al. 2019), principal component analysis (PCA) assisted BO (PCA-BO) (Raponi et al. 2020), kernel PCA assisted BO (KPCA-BO) (Antonov et al. 2022), and trust region BO (TuRBO) (Eriksson et al. 2019). A comprehensive analysis of these BO variants is available in Santoni et al. (2023).

Since the performance of aforementioned BBO algorithms is highly dependent on their hyperparameter settings, an optimal configuration is necessary for the best performance w.r.t. the allocated time and computational resources, which is commonly known as hyperparameter optimization (HPO). Nevertheless, this is challenging for practitioners with limited experience in HPO. Furthermore, existing HPO approaches usually demand a considerable function evaluation budget, and thus, are less practical for real-world expensive BBO problems. To the best of our knowledge, research about this topic is still lacking. As an effort in filling the gap, an idea toward automated HPO based on cheap-to-evaluate *representative functions* as *surrogates* of expensive BBO problems was recently proposed in Long et al. (2022, 2024). Inspired by this work, we are motivated to further investigate the potential of the proposed HPO approach for automotive crashworthiness optimization.

1.1 Our contributions

We propose an automated approach to optimally fine-tune optimization algorithms for BBO problems with expensive function evaluations. In brief, the optimization landscape of a BBO problem is analyzed using exploratory landscape analysis (ELA) to identify cheap-to-evaluate representative functions that belong to the same optimization problem class. Similar to prior predictive check (Gabry et al. 2019),

these representative functions can then be exploited to fine-tune algorithm configurations specifically for this BBO problem through HPO. Subsequently, optimal configurations can be identified at a significantly reduced computational cost compared to optimization runs on the BBO problem using expensive function evaluations.

Focusing on three state-of-the-art BBO algorithms, namely CMA-ES, DE, and BO, our results show that our approach has promising potential in identifying optimal configurations for the black-box optimization benchmarking (BBOB) functions, outperforming the readily available default configuration. When applied to an automotive side crash problem using two load cases, the optimal configurations identified using our approach are superior than the default configuration as well as the widely used RSM w.r.t. the best-found optimization solutions and convergence speed.

Ultimately, our goal is to assist practitioners in automatically and optimally fine-tuning optimization algorithms for their applications, and thereby, potentially accelerating development projects and/or lowering production cost. Beyond automotive crashworthiness optimization, our approach is designed to be applicable in other expensive BBO domains with adaptations.

This article has the following structure: We summarize related works in Sect. 2, our methodology in Sect. 3, and experimental setup in Sect. 4. This is followed by results and discussion in Sect. 5. Lastly, conclusions and future works are provided in Sect. 6.

2 Related work

Our approach is developed based on the findings in Long et al. (2022, 2024), namely (i) there exists a set of randomly generated functions (RGF) that belong to the same optimization problem class as automotive crashworthiness optimization, and (ii) these RGF are indeed informative in estimating the actual performance of different algorithm configurations on BBO problems. In short, we mainly improve their approach in the following extensions:

1. A selection process is integrated to identify RGF that are appropriate as representative functions for HPO purposes to identify optimal configurations;
2. Instead of considering only a limited set of configurations using a grid search approach, we employ optimizers for HPO to facilitate an automated exploration of the hyperparameter search space in searching for optimal configurations;
3. For a reliable identification of optimal configurations, two sets of representative functions are considered for HPO, similarly to the train-test split in a machine learning (ML) context; and
4. Moreover, on top of the BBOB suite, our approach is additionally evaluated based on an automotive side crash using two load cases.

2.1 Hyperparameter optimization

Due to the fact that heuristics optimization algorithms are sensitive to their hyperparameters, HPO plays a decisive role to achieve optimal performance. Correspondingly, different HPO approaches have been proposed, such as tree-structured Parzen estimator (TPE) (Bergstra et al. 2011) and sequential model-based algorithm configuration (SMAC) (Lindauer et al. 2022a), which can handle mixed-integer search space, consisting of continuous, discrete, categorical, and/or conditional variables. While having a similar framework as standard BO, instead of using GP models to approximate the search space, Parzen estimators are employed in TPE and random forest (RF) models in SMAC. For instance, Zhao and Li (2018) investigated the fine-tuning of three learning rates of CMA-ES using TPE on the BBOB suite.

2.2 Black-box optimization benchmarking suite

To fairly evaluate the performance of optimizers developed for HPO or BBO in general, the BBOB suite is one of the most utilized benchmark set (Hansen et al. 2010). Proposed by Hansen et al. (2009), the BBOB suite consists of altogether 24 single-objective, noiseless, and continuous functions, which we refer to as *the* BBOB in this work. To facilitate the benchmarking of different optimization algorithms, the BBOB suite has been integrated into the comparing continuous optimizers (COCO) platform (Hansen et al. 2021) and iterative optimization heuristics profiler tool (IOHProfiler) (Doerr et al. 2018).

Commonly, investigations based on the BBOB functions focus on the search space $[-5, 5]^d$ with the global optimum located inside $[-4, 4]^d$, where d represents the problem dimensionality. Not only being scalable to arbitrary dimensionality, the BBOB functions are beneficial in that different instances can be generated via transformations of the search space and objective values, which are controlled by an internal identifier. Detailed analysis of the BBOB instances is available in Long et al. (2023b).

2.3 Randomly generated functions

Apart from the carefully designed BBOB functions with known properties, Tian et al. (2020) introduced a function generator that can create tree-based RGF of various optimization complexity. Principally, a tree-structured function expression is constructed by combining a set

of mathematical operands and operators, which are randomly selected from a pre-defined pool using a selection probability.

In fact, it has been shown that RGF have a higher diversity in terms of optimization complexity compared to the BBOB functions (Škvorc et al. 2021a). Nevertheless, unlike the well-understood BBOB functions, the properties of RGF are not known, such as the global optimum and function complexity. Subsequently, an extension has been recently attempted on this function generator, that is using genetic programming to guide the function generation toward specific optimization complexity (Long et al. 2023a).

2.4 Exploratory landscape analysis

In this work, the optimization complexity of BBO problems is captured using ELA, one of the popular tools developed for fitness landscape analysis to better understand the effectiveness and behavior of optimization algorithms (Malan 2021). Basically, ELA was proposed to numerically quantify the high-level properties of continuous optimization problems, e.g., multi-modality, global structure, and separability, based on six classes of low-level features, consisting of y-distribution, level sets, meta-models, local searches, curvature, and convexity (Mersmann et al. 2010, 2011). Since then, additional feature classes have been introduced to complement these fundamental features, namely dispersion, nearest better clustering (NBC), PCA, linear model, and information content of fitness sequences (ICoFiS) (Kerschke and Trautmann 2019b; Kerschke et al. 2015; Lunacek and Whitley 2006; Muñoz et al. 2015a).

For ELA features computation, some DoE samples $\mathcal{X} = \{x_1, \cdots, x_n\}$ and the corresponding objective values $\mathcal{Y} = \{y_1, \cdots, y_n\}$ computed using an objective function f, i.e., $f : \mathcal{X} \rightarrow \mathcal{Y}$, are required as input, where $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, and n represents the DoE sample size. Following this, ELA features are dependent on the DoE sample size, sampling strategy, and dimensionality (Renau et al. 2019; Škvorc et al. 2021b; Muñoz et al. 2022). To overcome potential bias in the hand-crafted ELA features, van Stein et al. (2023) proposed to characterize BBO problems based on latent space features captured using deep neural networks (NN), which is beyond the scope of this work.

In fact, ELA features are indeed informative in estimating the performance of optimization algorithms in the so-called landscape-aware algorithm selection approach (Kerschke and Trautmann 2019a; van Stein et al. 2024). Principally, by training predictive models, e.g., using ML, the performance of different algorithm configurations for BBO problems can be estimated based on their ELA features (Benjamins et al. 2022). We refer to Muñoz et al. (2015b); Kerschke et al. (2019) for further information.

3 Methodology

The workflow of our approach is implemented partially based on Long et al. (2024) and visualized in Fig. 1, consisting of three main steps, namely (Step 1) identifying RGF that are appropriate as representative functions for HPO purposes, (Step 2) searching for optimal configurations through HPO using the cheap-to-evaluate representative functions, and (Step 3) applying optimal configurations identified to solve BBO problems using expensive function evaluations. In brief, optimal configurations that are identified based on some representative functions can be applied for solving an expensive BBO problem instance. Essentially, the HPO in Step 2 is first executed to identify optimal configurations using representative functions. Once completed, the optimal configurations are applied and kept unchanged throughout the actual optimization runs using expensive function evaluations in Step 3.

Notably, the *same* optimization setup required for actual optimization runs on BBO problems (Step 3) is defined for HPO (Step 2), e.g., using the same function evaluation budget and performance metric. Thus, the configurations identified are optimal specifically for this BBO problem and optimization setup. For a different setup or BBO problem classes, optimal configurations must be newly identified by re-running the pipeline.

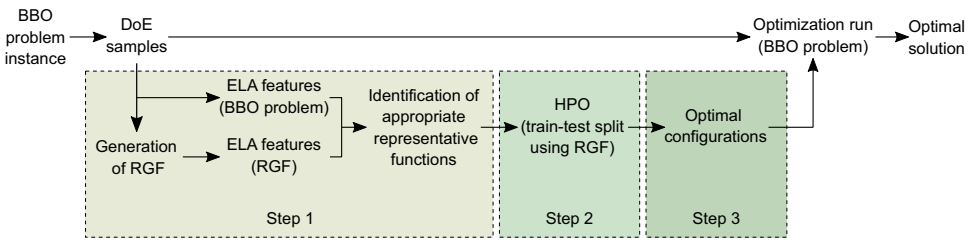


Fig. 1 An overview of the automated HPO pipeline proposed for expensive BBO problems, which can identify optimal configurations to solve a BBO problem instance efficiently w.r.t. time and computational resources. This is achieved by utilizing HPO based on some cheap-to-evaluate representative functions, e.g., RGF with similar optimization complexity as the BBO problem in terms of ELA features

Throughout this work, we refer to a configuration as a set of hyperparameter settings. Each aspect of the pipeline is described in detail in the following.

BBO problem instance

Firstly, a set of DoE samples of a BBO problem instance is required as input for our pipeline.

Generation of RGF

Secondly, a large set of RGF is generated and evaluated using the same DoE samples. For the function generation, we employ the function generator implemented in Long et al. (2024).

Computation of ELA features

Next, the ELA features of BBO problem instance and RGF are separately computed, where the objective values are beforehand normalized using min-max scaling to combat potential inherent bias in ELA features computation (Prager and Trautmann 2023a). To improve the applicability of our approach for expensive BBO problems, we consider only ELA features that can be computed using a DoE without requiring additional function evaluations, as summarized in Table 1. In cases where a feature computation fails, e.g., the DoE sample size is too small for level sets or linear model features, such feature is skipped. Due to the fact that some of the ELA features are redundant (Renau et al. 2019), highly correlated ELA features according to Pearson's correlation coefficient (> 0.95) are discarded.

Identification of representative functions

The similarity between BBO problem instance and RGF is measured based on the differences in ELA features quantified using Euclidean distance. Using an equal weighting of all ELA features, the Euclidean distance is computed based on standardized ELA features, i.e., removal of mean and scaling to unit variance. Accordingly, we can rank the RGF, where RGF having the smallest difference is considered as the most similar function.

Nevertheless, before the RGF can be considered as representative functions for HPO purposes, two challenges must be overcome, namely (i) the properties of RGF are not known a priori, e.g., the global optimum, and (ii) some RGF are insufficiently discriminative in distinguishing different configurations. For a clear explanation, we show the HPO results on three chosen RGF in Fig. 2. Generally, we consider functions similar to RGF1 appropriate for HPO purposes, where a clear ranking of configurations is possible with only a few ties and optimal configurations can be easily identified. On the other hand, RGF2 shows an extreme situation, where too many or all configurations are equally competitive in finding the same solution, leading to an ambiguous configuration ranking and difficulty in selecting optimal configurations. We suspect that such RGF have a low optimization complexity and consider them as incompetent for HPO purposes. It is worth noting that two RGF can exhibit completely opposite patterns in HPO results, despite having small differences in ELA features, which remains an open question for future work. Furthermore, to improve the reliability of HPO, we exclude functions similar to RGF3, where the best solution found is an extreme outlier that can only be occasionally found.

Consequently, the following steps are implemented to identify RGF that are appropriate as representative functions for HPO purposes.

1. The global optimum of RGF is estimated in a brute-forcing fashion, that is, by applying HPO using TPE and modular CMA-ES on each RGF. In such way, the global optimum y_{opt} of RGF is computed based on the best solution found during the HPO y_{hpo} using Eq. 1.

Table 1 In this work, the optimization complexity of BBO problems is quantified using the following 53 ELA features

Feature class	ELA feature
y-distribution (3 features)	`skewness`
	`kurtosis`
	`number_of_peaks`
Level set (9 features)	`mmce_lda_{10,25,50}`
	`mmce_qda_{10,25,50}`
	`lda_qda_{10,25,50}`
Meta-model (9 features)	`lin_simple.{adj_r2,intercept}`
	`lin_simple.coef.{min,max,max_by_min}`
	`lin_w_interact.adj_r2`
	`quad_simple.{adj_r2,cond}`
	`quad_w_interact.adj_r2`
Dispersion (16 features)	`ratio_mean_{02,05,10,25}`
	`ratio_median_{02,05,10,25}`
	`diff_mean_{02,05,10,25}`
	`diff_median_{02,05,10,25}`
NBC (5 features)	`nn_nb.{sd_ratio,mean_ratio,cor}`
	`dist_ratio.coeff_var`
	`nb._fitness.cor`
Linear model (4 features)	`avg_length.{reg,norm}`
	`length.mean`
	`ratio.mean`
PCA (2 features)	`expl_var_PC1.{cov_init,cor_init}`
ICoFiS (5 features)	`h.max`
	`eps.{s,max,ratio}`
	`m0`

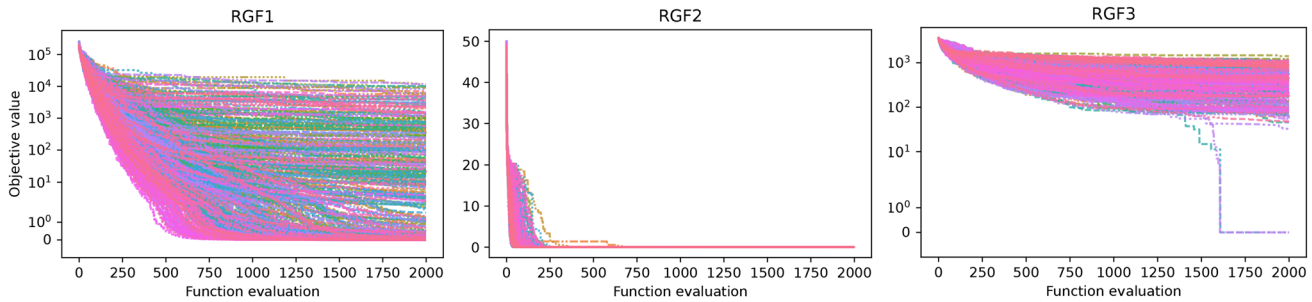


Fig. 2 The HPO results for three chosen RGF, showing the optimization convergence curves of altogether 500 configurations. Each curve represents the optimization convergence of a modular CMA-ES configuration (median over 20 repetitions). The function evaluation is shown on the x-axis and the re-scaled objective value on the y-axis, with 0 being the best solution found in all runs. A smaller objective value is better. (*Left*) A clear configuration ranking, where optimal configurations can be identified. (*Middle*) An ambiguous configuration ranking, where all configurations are equally competitive. (*Right*) The best solution found during HPO is an outlier that can only be occasionally found

$$y_{opt} = \begin{cases} \lfloor y_{hpo} \rfloor, & \text{if } 0 \leq \left| y_{hpo} \right| < 10 \\ \lfloor y_{hpo}/10 \rfloor \cdot 10, & \text{if } 10 \leq \left| y_{hpo} \right| < 100 \\ \lfloor y_{hpo}/10^p \rfloor \cdot 10^p, & \text{otherwise} \end{cases}, \quad (1)$$
$$p = \left\lfloor \log_{10} \left| y_{hpo} \right| \right\rfloor - 1,$$

where y_{hpo} is either rounded to the nearest lower integer for a small $\left| y_{hpo} \right|$, or rounded based on the nearest lower power of 10. Having an estimated global optimum of RGF is crucial in our approach to facilitate an evaluation and comparison of configurations between different representative functions with varying scale ranges. The detailed setup of HPO is described in Sect. 4.

2. To determine the ambiguity of configuration ranking, all configurations evaluated in the HPO are ranked according to their performances, and compared against a strict ranking (without tie). Based on Kendall rank correlation coefficient, a configuration with a correlation lower than 0.9, e.g., due to too many ties, is considered as ambiguous.
3. Moreover, the global optimum is considered as an outlier, if it is greater than 3 standard deviations away from the mean distribution of HPO results based on the standard score or z-score.
4. If any of these conditions (point 2 or 3) is fulfilled, such RGF is considered inappropriate for HPO purposes and eliminated.

HPO with training–testing split

To enhance the generalization of optimal configurations identified across a particular optimization problem class, which is similar to the over-fitting problem in ML, several representative functions are considered for HPO.

Precisely, the representative functions are split into two groups of training and testing functions for HPO. During HPO, the mean performance of each configuration is evaluated using all training functions, where the objective values of different training functions are min-max normalized according to their global optimum and worst DoE samples. Afterward, several top-performing configurations are re-evaluated using the testing functions, where the re-evaluated configurations will be ranked according to their performance to identify optimal configurations for the problem class.

In this work, the top 10% configurations on the training functions identified using HPO are re-evaluated on the testing functions and ranked according to their new performances. While an ideal ratio of training–testing functions remains to be investigated, two training functions and one testing function are considered in this work, which can perform well on average in our preliminary testing.

Optimization run

Lastly, optimal configurations or the best configuration identified can be applied to solve the BBO problem instance.

Despite additional computational resources are needed for the HPO in our approach, we argue that (i) the optimal configurations identified can improve the efficiency in solving BBO problems and (ii) the overall HPO cost is relatively minor compared to the actual optimization runs on BBO problems. A brief comparison of the function evaluations between HPO and solving an expensive BBO problem is provided in Table 2, using automotive crash optimization as a representative example. In short, for the identification of optimal configurations (Step 1 to Step 3 in Fig. 1), the overall computational cost is insignificant compared to the crash optimization. Moreover, by re-using the same representative functions and optimal configurations identified, these steps can be potentially skipped for future BBO problems from the same problem classes. In this regards, a detailed run time analysis is planned for future work.

Table 2 A rough estimation of the computational cost required for the HPO in our approach and a typical crashworthiness optimization problem in the automotive industry. Depending on the problem definition and optimization setup, the computational cost can be different for other BBO problems

Estimation	Hyperparameter optimization	Automotive crash optimization
Function evaluation	Representative function	FE simulation
Cost per function evaluation	Cheap: Within seconds (using 1 CPU core)	Expensive: > 24 hours (using 192 CPU cores)
Total wall clock time	A few hours	Several days / weeks

4 Experimental setup

For a comprehensive assessment, the performance of our approach is evaluated based on the BBOB suite and an automotive side crash as a representative real-world expensive BBO problem.

4.1 BBOB functions

Generally, the experimental setup for the BBOB functions can be summarized as follows.

- 24 BBOB functions of the first instance in $20d$, which is a typical problem dimensionality for automotive crash optimization.
- Representative functions are selected from a large set of 10 000 RGF.
- The ELA features summarized in Table 1 are computed using `pflacco` (Prager and Trautmann 2023b) based on a DoE of $20 \cdot d$ samples and Sobol' sampling strategy (Sobol' 1967), as suggested in Renau et al. (2020).
- Focusing on fine-tuning the configurations of modular CMA-ES (de Nobel et al. 2021), modular DE (Vermetten et al. 2023), and three BO variants, consisting of vanilla BO (or simply BO), TuRBO-1, and TuRBO-m.
- For modular CMA-ES and DE, each configuration is allocated with $100 \cdot d$ evaluations and 20 repetitions, while a smaller budget of 250 evaluations for BO variants, as the time-complexity of training GP models increases exponentially with sample size.
- For modular CMA-ES and DE, TPE available in `HyperOpt` (Bergstra et al. 2013) and SMAC (Lindauer et al. 2022b) are employed for HPO, with a budget of 500 configuration evaluations. On the other side, only TPE and 50 evaluations are utilized for BO variants due to the smaller hyperparameter search space (refer to Sect. 4.3).
- For modular CMA-ES and DE, our approach is compared against the default configuration, single best solver (SBS), and virtual best solver (VBS), as introduced in Sect. 4.4. On the other hand, we compare only against the default configuration for BO variants, since they are computationally intensive.

4.2 Automotive side crash

In this work, we focus on a side crash scenario as a representative automotive crash problem, as shown in Fig. 3. Using a FE submodel developed based on state-of-the-art modeling of BEV, the vehicle is impacted from the sideways against a rigid pole at a speed of 32 km/h and at a small angle, following the guidelines regulated by the European New Car Assessment Program (Euro NCAP 2023). Altogether two load cases are investigated, which are described further in Sect. 4.2.1 and Sect. 4.2.2. In this work, our crash problems are defined as unconstrained single-objective optimization problems, which requires further investigations and improvements in the future (Sect. 6).

Each FE simulation is terminated after 50 ms and requires a computation time of roughly 6 minutes using the explicit solver `LS-DYNA` (Livermore Software Technology Corporation 2019) in its MPP version distributed across 64 CPU cores in high-performance computing clusters. Altogether, the FE submodel consists of 95 000 elements. Regarding the design variables, our main focus is to optimize 18 thicknesses of rocker panels (made of aluminum alloys) and supporting beams (made of high-strength steels).

Generally, the mass m and structural performance of vehicle design are two crucial objectives in automotive crash optimization. For the side crash problem investigated, the structural performance is represented by the maximum structural deformation toward the battery compartment, which we call intrusion u. Since both objectives are conflicting, e.g., minimizing the mass though a reduction in thicknesses usually leads to a weaker performance and larger intrusion, finding an acceptable trade-off is challenging. For the crash optimization, we focus on vanilla BO due to its superior performance in our preliminary tests, and compare it against the default configuration and RSM (refer to Sect. 4.4). For reference, the vehicle design located at the center of design space has a total component mass of 32.4 kg.

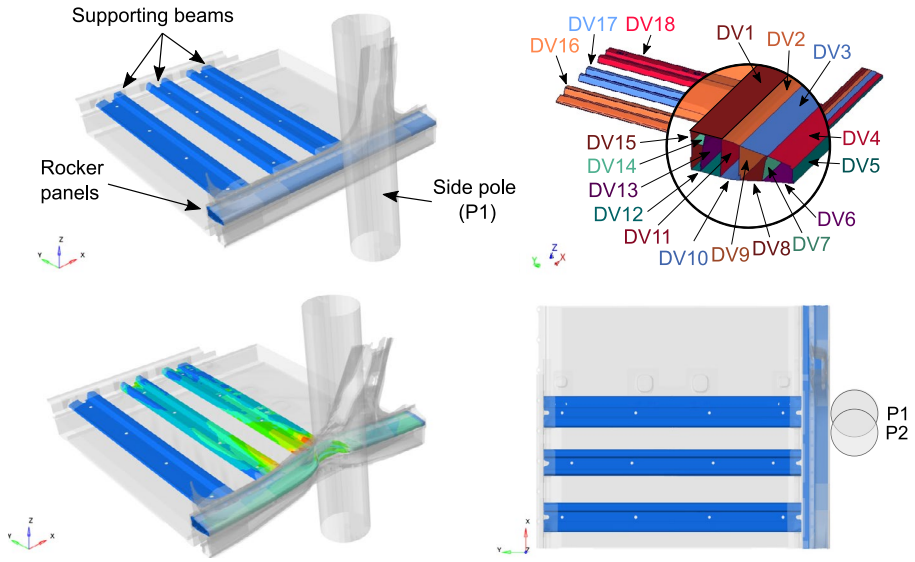


Fig. 3 An overview of the FE submodel developed for automotive side crash scenario against a rigid pole. In BEV, the battery compartment is typically installed underneath the supporting beams and between the rocker panels on both sides. (*Top left*) The thicknesses of rocker panels and supporting beams are the design variables considered for optimization, as highlighted in blue color. (*Bottom left*) An example of FE simulation result, showing the deformation and plastic strain distribution in the submodel during impact. (*Top right*) In total, 18 design variables in the rocker panels (DV1 – DV15) and supporting beams (DV16 – DV18), where each design variable is highlighted using a different color. (*Bottom right*) Top view of the submodel, showing two pole positions (P1 and P2) considered for the crash problem

4.2.1 Load case A—Single-pole impact

In the first load case or *single* pole impact, the vehicle is impacted against the rigid pole at position P1 (refer to Fig. 3), using the optimization objective f_{opt} defined in Eq. 2.

$$\begin{aligned} \min f_{opt} &= m' + u', \\ m' &= \frac{m - m_{min}}{m_{max} - m_{min}}, \\ u' &= \frac{u - u_{min}}{u_{max} - u_{min}}, \end{aligned} \tag{2}$$

where m' and u' are the min-max normalized mass and intrusion based on the minimum and maximum of DoE. Using a similar setup as for the BBOB functions, we consider a DoE of fixed 400 samples for the ELA features computation and optimization, 30 re-samplings using FE simulations, and five optimization repetitions.

4.2.2 Load case B—Multi-pole impact

In real-world situation, a side impact can occur at any position alongside the vehicle body, which is difficult to be precisely pinpointed. Following this, vehicle design is often optimized w.r.t. *multiple* pole impacts, or *multi*-pole impact, aiming to identify a robust design for all impact scenarios considered. Subsequently, solving a multi-pole crash problem is challenging, because the structural performance of a vehicle design can vary strongly depending on the impact positions.

Using the same FE submodel, we investigate the multi-pole crash problem based on pole position P1 and P2 (refer to Fig. 3) using the objective f_{opt} defined in Eq. 3.

$$\min f_{opt} = \alpha \cdot m' + u'_1 + u'_2, \tag{3}$$

where $\alpha = 2$ for an equal weighting of mass and intrusions and $u'_{i\in\{1,2\}}$ represents the intrusion for position P1 or P2 using the same normalization as in Eq. 2.

Since each vehicle design in multi-pole crash problems must be separately evaluated using a FE simulation for each pole position, each DoE sample and optimization re-sampling requires two FE simulations. Consequently, a smaller setup is considered to minimize the computational effort, namely a DoE of 200 samples, 30 re-samplings, and three optimization repetitions.

4.3 Performance metric and search spaces for HPO

In many real-world applications, having a faster optimization convergence, i.e., finding acceptable solutions in a shorter time or using less computational resources is often as important as finding the global optimum. In fact, due to the highly nonlinear nature of crashworthiness problems, identifying

the global optimum is usually impractical or even impossible in industrial crash applications.

Following this, we propose to evaluate the performance of a configuration based on its area under the curve (AUC) of optimization convergence (Fig. 2), which is informative about the solution found and convergence speed. By minimizing the AUC metric, we are essentially looking for configurations that have an optimal balance between vehicle design and speed. In this work, the AUC is computed using min-max normalized objective values, which are based on the global optimum and the worst DoE sample, and then it is divided by the evaluation budget.

In the meantime, the hyperparameter search space for modular CMA-ES is summarized in Table 3, modular DE in Table 4, and BO variants in Table 5.

4.4 Optimization baseline

For a fair evaluation of the optimal configurations identified using our approach, the following configurations are considered as comparison baseline.

Default: The readily available algorithm configuration in its proposed implementation that is simply taken off-the-shelf and directly used by practitioners unfamiliar with HPO. In line with our motivation, this is our primary comparison baseline.

SBS: The configuration that can perform well on average for a set of BBO problems, e.g., the BBOB suite, which is identified through HPO. This configuration serves as our secondary comparison baseline in this work.

VBS: The best performing configuration for a particular BBO problem identified using HPO, which can be treated as the lower bound.

On the other hand, RSM and SRSM are considered as benchmark references for automotive crashworthiness

Table 3 An overview of the hyperparameters of modular CMA-ES considered in this work. The default configuration is highlighted in bold, where the default learning rates are automatically computed depending on other hyperparameters

Hyperparameter	Type	Search space
Number of children λ	Integer	{ 5,..., 50 } (**4** + ⌊(**3ln**(**d**))⌋)
Number of parent μ (as ratio of λ)	Continuous	[0.3, 0.5] (**0.5**)
Initial standard deviation σ_0	Continuous	[0.1, 0.5] (**0.2**)
Learning rate step size control	Continuous	[0.0, 1.0]
Learning rate covariance matrix adaptation	Continuous	[0.0, 1.0]
Learning rate rank-μ update	Continuous	[0.0, 0.35]
Learning rate rank-one update	Continuous	[0.0, 0.35]
Active update	Categorical	{ True, **False** }
Mirrored sampling	Categorical	{ **none**, 'mirrored', 'mirrored pairwise' }
Threshold convergence	Categorical	{ True, **False** }
Recombination weights	Categorical	{ '**default**', 'equal', '1/2^lambda' }

Table 4 An overview of the hyperparameters of modular DE considered in this work. The default configuration is highlighted in bold

Hyperparameter	Type	Search space
Population size λ	Integer	{ 5,..., 50 } (**4** + ⌊(**3ln**(**d**))⌋)
Weighting factor F	Continuous	[0.0, 1.0] (**0.5**)
Crossover constant CR	Continuous	[0.0, 1.0] (**0.5**)
Base vector	Categorical	{ '**rand**', 'best', 'target' }
Reference vector	Categorical	{ **none**, 'pbest', 'best', 'rand' }
Number of differences	Categorical	{ **1**, 2 }
Weighted F	Categorical	{ True, **False** }
Crossover method	Categorical	{ '**bin**', 'exp' }
Eigenvalue transformation	Categorical	{ True, **False** }
Adaptation of F	Categorical	{ **none**, 'shade', 'shade-modified', 'jDE' }
Adaptation of CR	Categorical	{ **none**, 'shade', 'jDE' }
Use JSO caps for F and CR	Categorical	{ True, **False** }

Table 5 An overview of the hyperparameters for the three BO variants considered in this work. The default configuration is highlighted in bold. The hyperparameter DoE size is defined as ratio of the evaluation budget. For investigations on the automotive crash problem, the DoE size is fixed and not considered for HPO

Variant	Hyperparameter	Type	Search space
BO	DoE size	Continuous	[0.1, 0.9] (**0.5**)
	Kernel of GP models	Categorical	{ Matérn 3/2, **Matérn 5/2**, RBF }
	Acquisition function	Categorical	{ **Expected improvement (EI),** Probability of improvement (PI), Lower confidence bound (LCB) }
TuRBO-1	DoE size	Continuous	[0.1, 0.9] (**0.5**)
	Infill points	Integer	{ 1,..., 10 } (**1**)
TuRBO-m	DoE size	Continuous	[0.1, 0.9] (**0.5**)
	Number of trust regions	Integer	{ 2,..., 6 } ($\lfloor$(**d**/**5**)$\rfloor$)
	Infill points	Integer	{ 1,..., 10 } (**1**)

optimization problems, using the performance of RSM as our baseline.

RSM: Based on some DOE samples, seven types of surrogate model are constructed, namely polynomial, support vector regressor, RF, gradient boosting, dense NN, GP, and k-nearest neighbor (Pedregosa et al. 2011), where the model hyperparameters are optimized using TPE based on a framework similar to Komer et al. (2014) and a five-fold cross-validation. Using the best-fitted surrogate model to approximate the crash problems, modular CMA-ES is then applied to search for the global optimum.

SRSM: In this work, optimization runs using SRSM are performed based on a similar approach described in Raponi et al. (2021); Stander and Craig (2002a, 2002b); Kok and Stander (1999). The response surfaces are constructed in the same way as the aforementioned RSM, where the same total function evaluation budget (DoE + re-samplings) is equally divided across five iterations. Moreover, DoE samples from previous iterations that are within the current subspace are additionally included for the construction of response surfaces.

5 Results and discussion

5.1 Reproducibility

To keep this article within a reasonable length, essential experimental results and figures that are not included are made available in our repository at https://zenodo.org/records/11350771.

5.2 Visualization of representative functions for BBOB

Before delving into analyzing the HPO results, the RGF identified as representative functions based on ELA features are visually inspected. Particularly, we compare the optimization landscape topology between the BBOB functions in 2*d* and their corresponding representative functions, which can be easily visualized compared to the high-dimensional automotive crash problems. In fact, the representative functions indeed have a similar optimization landscape for many BBOB functions, such as F1 (Sphere), F22 (Gallagher's Gaussian 21-hi Peaks), and F23 (Katsuura), as presented in Fig. 4.

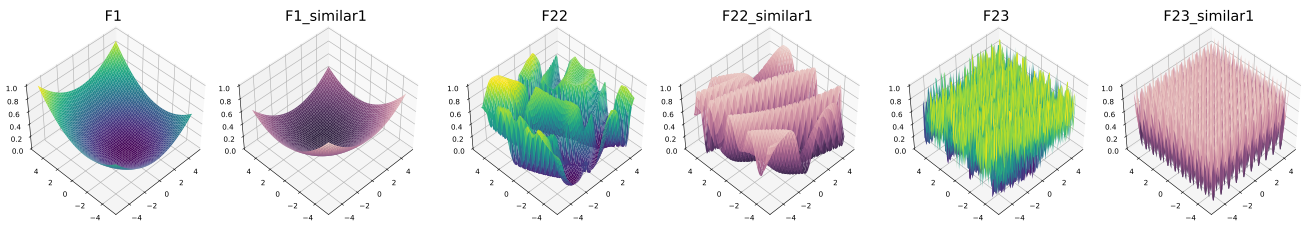


Fig. 4 Visual comparison of optimization complexity between pairs of selected BBOB functions (F1, F22, and F23) in 2*d* (*left; yellow-green color*) and their corresponding representative functions identified based on ELA features (*right; beige-purple color*). The search space $[-5, 5]^2$ is shown on the in-plane axis, while the min-max normalized objective values [0, 1] are on the vertical axis, with 0 being the global optimum. A lighter color represents a larger objective value, while a darker color for a smaller value. Refer to Sect. 5.1 for full results

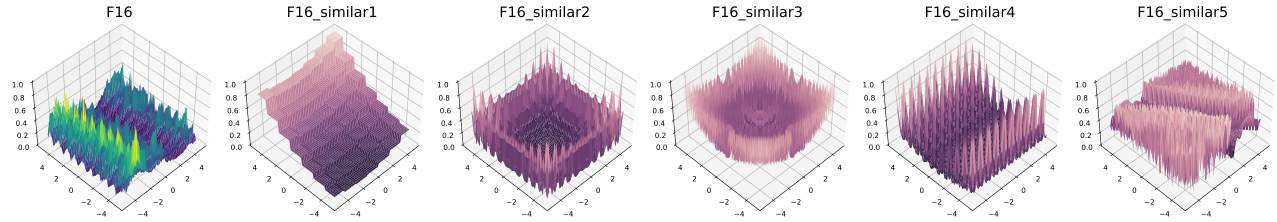


Fig. 5 Visual comparison of optimization complexity between F16 in 2*d* and the first five RGF with the most similar landscape characteristics. The RGF are labeled from `similar_1` to `similar_5` in ascending order according to their differences in ELA features. The search space $[-5, 5]^2$ is shown on the in-plane axis, while the min-max normalized objective values [0, 1] are on the vertical axis, with 0 being the global optimum, using the same color scheme as in Fig. 4

Nevertheless, for some complex functions like F16 (Weierstrass), visual differences in the optimization landscape of representative functions can be observed, as illustrated in Fig. 5. Interestingly, apart from the first RGF (`similar_1`), the optimization landscape of the other four RGF is much more similar to F16, that is, highly rugged and repetitive landscape. While an explanation remains yet unclear, we suspect that (i) this might be due to our approach in selecting and processing the ELA features, and (ii) ELA features that can properly capture such complex landscape characteristics are still lacking. Due to the fact that the differences in their ELA features are rather small, we incline toward the first assumption that perhaps an equal weighting of all ELA features for the distance computation is not optimal for all BBO problems (Sect. 3). At the same time, this highlights the importance of considering multiple representative functions for HPO using a training–testing split in our approach.

5.3 HPO for BBOB functions

Apart from visual inspections, the optimization performances using different configurations of modular CMA-ES, modular DE, and BO variants are analyzed and compared in Fig. 6, using F1 and F10 (Ellipsoidal) in 20*d* as an example. Similar to F1, we can observe that the optimal configurations identified using our approach have a smaller AUC than the default configuration for most BBOB functions, revealing that our approach is superior than the default configuration, even for complex functions like F16 and F23. Furthermore, our approach can outperform the SBS for several BBOB functions, while being equally competitive as the SBS for a few of the remaining BBOB functions. This is particularly encouraging for real-world applications, as such a SBS is typically not available. Nonetheless, there is still room for improvement in our approach, as we struggle on a few BBOB functions like F10, as shown in Fig. 6, where the

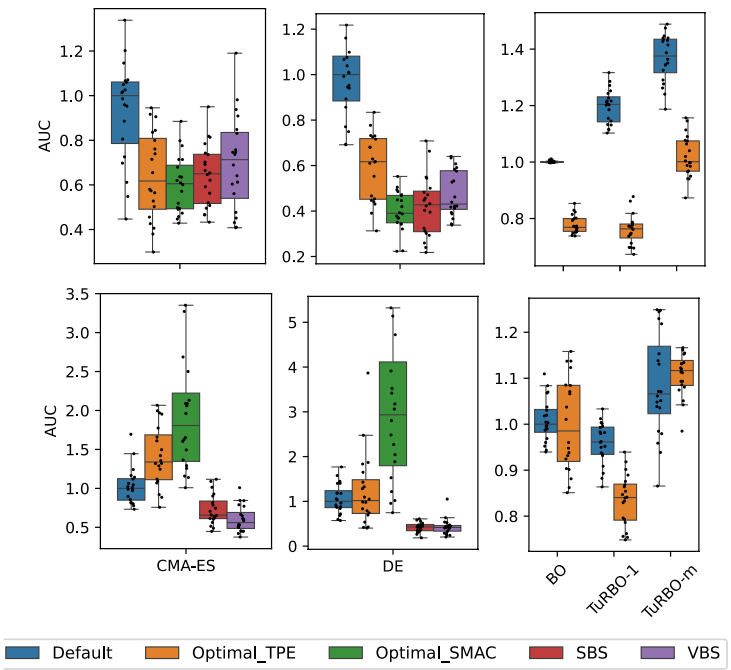


Fig. 6 Performance of modular CMA-ES (*left*), modular DE (*middle*), and BO variants (*right*) using different configurations for F1 (*top row*) and F10 (*bottom row*) in 20*d*, with a repetition of 20 times for each configuration. The AUC is computed based on the min-max normalized objective values (according to the global optimum and worst solutions in all runs) and divided by the evaluation budget. To allow a convenient interpretation across different functions, here the AUC is additionally divided by the median AUC of the default configuration. A configuration with a smaller AUC is better. (*Legend*) Default configuration, optimal configurations identified using TPE or SMAC, SBS, and VBS. Refer to Sect. 5.1 for full results

performance of optimal configurations identified is clearly weaker than the default configuration and SBS.

In general, similar observations can be made for modular DE and BO variants, showing an attractive potential of our approach in solving the BBOB functions. Further comparisons indicate that our approach tends to work more reliably with modular CMA-ES than with modular DE, based on the fact that optimal configurations of modular CMA-ES can outperform the default configuration and compete with the SBS across a larger set of BBOB functions. The poor performance of modular DE might be partly attributed to its stochastic nature, e.g., randomness in population initialization, which warrants further investigations. On the other hand, our approach seems to be working well with BO variants, where a smaller DoE size is favored in all optimal configurations identified.

For a precise analysis, the performance of different configurations is statistically evaluated based on the Wilcoxon signed-rank test in `scipy` (Virtanen et al. 2020), using the alternative hypothesis *the performance of optimal configurations identified by our approach is weaker*. The statistical results for all 24 BBOB functions using modular CMA-ES, modular DE, and BO variants are presented in Fig. 7. In line with our previous observations, optimal configurations identified using our approach can in fact outperform the default configuration for many BBOB functions and compete against the SBS in some cases. While not being the focus of this work, SMAC seems to have slight advantage in finding better configurations compared to TPE. Nevertheless, our current approach does not work for several BBOB functions, such as F7 (Step Ellipsoidal), F10 (Discus), and F11, which might be related to the processing of ELA features, as discussed in Sect. 5.2. Overall, these results reveal the potential of our approach in generalizing to a wide range of optimization problem classes within the BBOB suite.

5.4 HPO for automotive side crash

In the following, the optimization results for the automotive side crash problem are presented and discussed.

5.4.1 Load case A – Single-pole impact

For the automotive side crash problem with single-pole impact, better vehicle designs with smaller objective values can be found by RSM, SRSM, and BO compared to the classical one-shot optimization strategy, as illustrated in Fig. 8. Precisely, SRSM has the best performance, followed by BO, which can generally outperform RSM in finding better vehicle designs. While not included in this work, increasing the DoE size by an additional 200 samples (+50%) does not clearly improve the optimization results for RSM in our testing, suggesting that RSM indeed struggles to approximate the complex side crash problem accurately. On the other side, the best overall vehicle design can be identified using SRSM, showing the advantage of an iterative optimization process to solve automotive crash problems. Meanwhile, the optimal BO configuration identified can perform slightly better than the default BO configuration in terms of the best vehicle design found (roughly −2%) and AUC metric

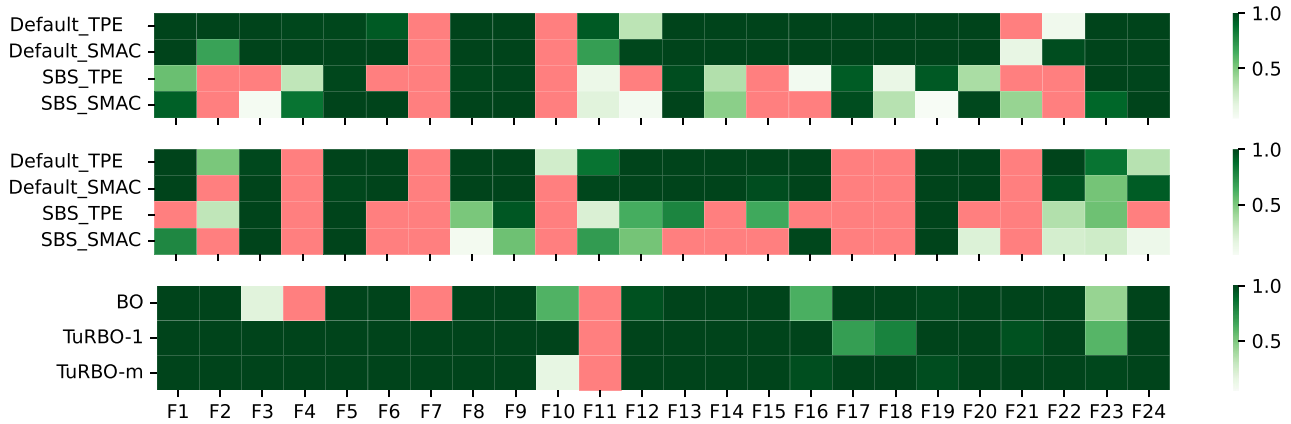


Fig. 7 Pairwise performance comparison between the best configuration found using TPE or SMAC in our approach against the default configuration and SBS of modular CMA-ES (*top*), modular DE (*middle*), and three BO variants (*bottom*) for 24 BBOB functions. Based on the Wilcoxon signed-rank test, a red color indicates that there is a statistically significant evidence to support the alternative hypothesis *the performance of our optimal configurations is weaker*, with a p value smaller than 0.05. On the other hand, a green color indicates that our optimal configurations are equally competitive or better, where a darker green color (higher p value) indicates that the hypothesis is less likely to be rejected. (*Legend*) Comparison of the default configuration or SBS against optimal configurations identified using TPE or SMAC

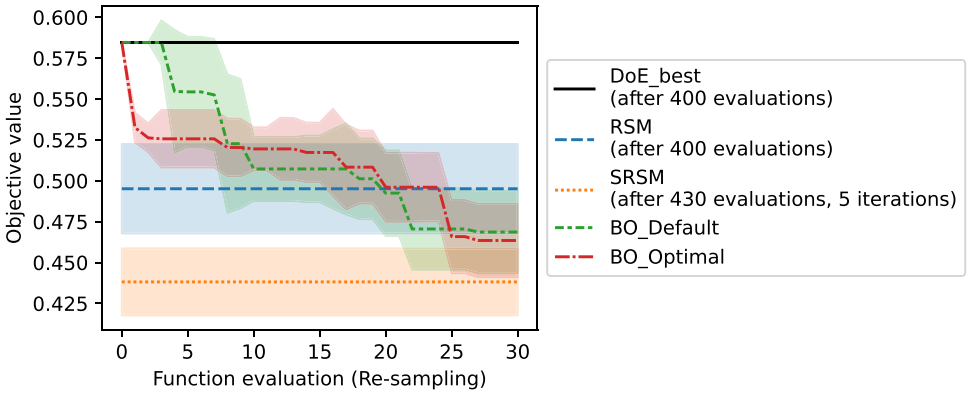


Fig. 8 Comparison between the one-shot optimization strategy (*black*), RSM (*blue*), SRSM (*orange*), default BO configuration (*green*), and optimal BO configuration predicted using our approach (*red*) for the side crash problem with single-pole impact (Eq. 2). For RSM and SRSM, the best-found solution is presented. On the other hand, the best-so-far solution during re-sampling is shown for BO, starting with the best DoE sample. The median performance and standard deviation of five repetitions are reported. A smaller objective value and lower AUC is better

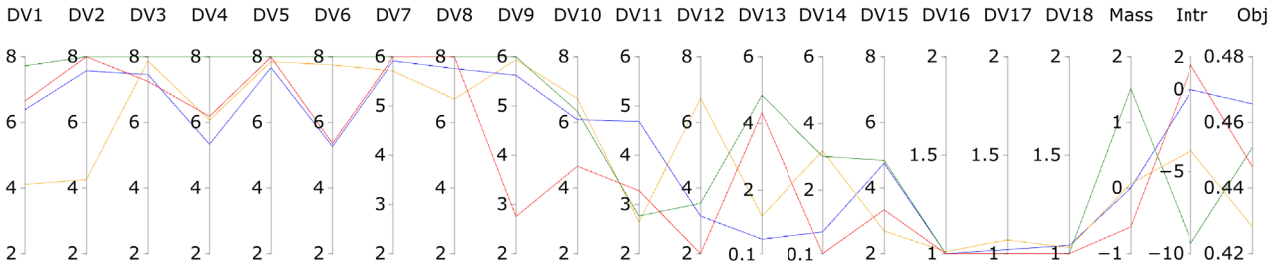


Fig. 9 Comparison of vehicle designs for the side crash problem with single-pole impact found by RSM (*blue*), SRSM (*orange*), default BO configuration (*green*), and optimal BO configuration identified using our approach (*red*), with the upper and lower bound of design space (thickness in mm) shown at the top and bottom row. The relative differences in mass (kg) and intrusion (mm) w.r.t. the vehicle design provided by RSM are shown. The optimization objective value is presented in the last column

(roughly −3%), having a faster convergence particularly at the beginning of optimization.

Next, the vehicle designs found by RSM, SRSM, and BO are analyzed further w.r.t. design space, as visualized in Fig. 9, focusing on the best optimization run with the smallest objective value among all repetitions. In brief, a general tendency can be observed for most vehicle designs, where a combination of thick rocker panels for regions close to the impact (DV1 – DV8) and thin supporting beams (DV16 – DV18) is favorable. On the contrary, the thicknesses of the remaining design variables on the inner side of rocker panels (DV9 – DV15) are more flexible.

Apart from that, the vehicle designs found by the optimization algorithms are different optima. Using RSM as a reference, the vehicle design provided by SRSM has a minimal increase in mass (+0.1 kg), but a good reduction in intrusion (−3.7 mm). On the other hand, while the default BO configuration favors a reduction in intrusion by increasing the thicknesses of rocker panels, our approach attempts to minimize both the vehicle mass and intrusion. In this context, the vehicle design found by the default BO configuration exhibits a reduction in intrusion (−9.4 mm) at the cost of an increased mass (+1.5 kg). Conversely, the vehicle design provided by our approach shows the opposite effect, having a smaller mass (−0.6 kg), but larger intrusion (+1.5 mm). The intrusions for different vehicle designs are visualized in Fig. 10, where the vehicle design found using our approach indeed deforms to a greater extent compared to the default BO configuration.

5.4.2 Load case B—Multi-pole impact

For the side crash problem with multi-pole impact, the optimization results are analyzed in a similar fashion as in Sect. 5.4.1 and presented in Fig. 11. As expected, better vehicle designs with smaller objective values can be found using RSM, SRSM, and BO compared to the one-shot optimization strategy. Unlike the previous single-pole load case, here SRSM performs worse than RSM in finding a better vehicle design, despite having the possibility to discover better solutions over iterations. In

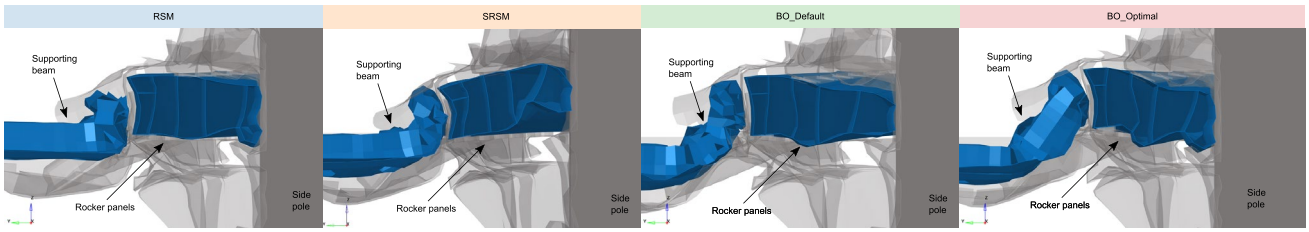


Fig. 10 The cross-sections of FE submodel, revealing the structural deformation during impact for the vehicle design found by RSM, SRSM, default BO, and optimal BO configuration identified using our approach for an impact at pole position P1. The same setting is applied for all simulation snapshots, e.g., deformation scaling, view position, and time frame

comparison to the previous load case, a smaller total function evaluation budget is allocated for the multi-pole load case. Thus, we believe that the complex multi-pole crash function is poorly approximated in SRSM due to the smaller DoE sample size. Meanwhile, the default BO configuration appears to be stuck in a local optimum, having a somewhat comparable performance as SRSM. On the contrary, better vehicle designs can be clearly found using optimal BO configurations identified through our approach, outperforming both RSM and default BO configuration. In fact, the optimal BO configuration has a comparable convergence speed as the default BO configuration based on AUC metric (roughly +1%), but can find a much better vehicle design (roughly −10%).

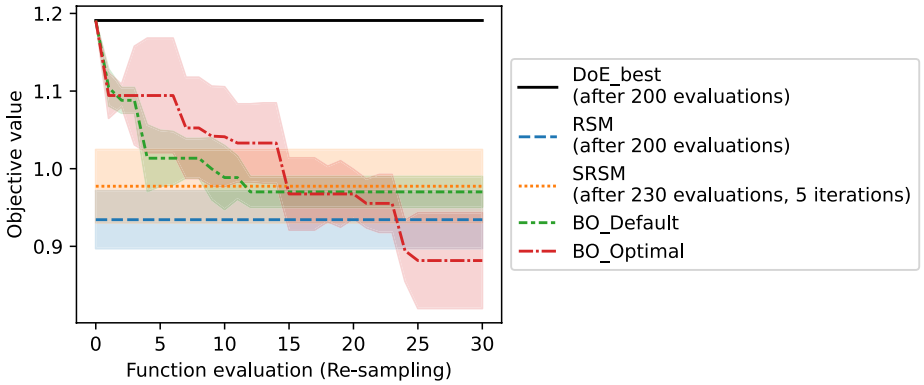
Regarding the design space, a similar trend in vehicle design as previously in the single-pole impact can be observed here, where thick rocker panels and thin supporting beams are favorable. Basically, most of the optimization algorithms push the vehicle design toward the boundary of design space, indicating that thick rocker panels could improve the overall structural performance of vehicle design that is robust against impacts at both pole positions P1 and P2. On the other side, our optimal BO configuration can better explore the search space, where the vehicle design has a combination of thinner rocker panels and slightly thicker supporting beams. Using RSM as a baseline, the vehicle design identified using SRSM has a slightly lower mass (−0.1 kg), but larger intrusions at both P1 (+0.9 mm) and P2 (+5.6 mm). Meanwhile, the vehicle design found by the default BO configuration is lighter (−0.5 kg), but at a cost of larger intrusions for an impact at P1 (+3.1 mm) and P2 (+5.2 mm). In comparison, the vehicle design provided by our approach is arguably better, having a lower mass (−0.7 kg), smaller intrusion for P1 (−0.5 mm), and slight increase in P2 (+1.8 mm), which is generally much more appealing.

It is worth noting that the optimization performance can be different, depending on the definition of optimization objectives, e.g., Eq. 2 and 3, which requires further investigations. Despite the findings of this work are limited to unconstrained single-objective optimization, we show that improvements in optimization performance can be gained using our approach compared to conventional method, in line with our motivations.

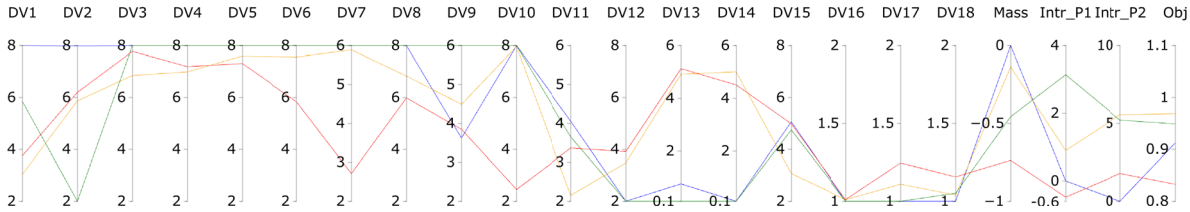
6 Conclusions and future research

An efficient solving of real-world expensive BBO problems, such as automotive crashworthiness optimization, at a minimal time and/or costs is critical, yet challenging. To overcome this problem, we propose an automated approach that can assist practitioners in optimally fine-tuning optimization algorithms for their applications. By exploiting cheap-to-evaluate RGF as representative functions, which belong to the same optimization problem class as BBO problems based on ELA features, optimal algorithm configurations can be identified using HPO at a significantly lower cost compared to the expensive function evaluations. To improve the reliability of our approach, we integrate a selection process to identify a set of RGF that are appropriate as representative functions and a train-test split for HPO purposes.

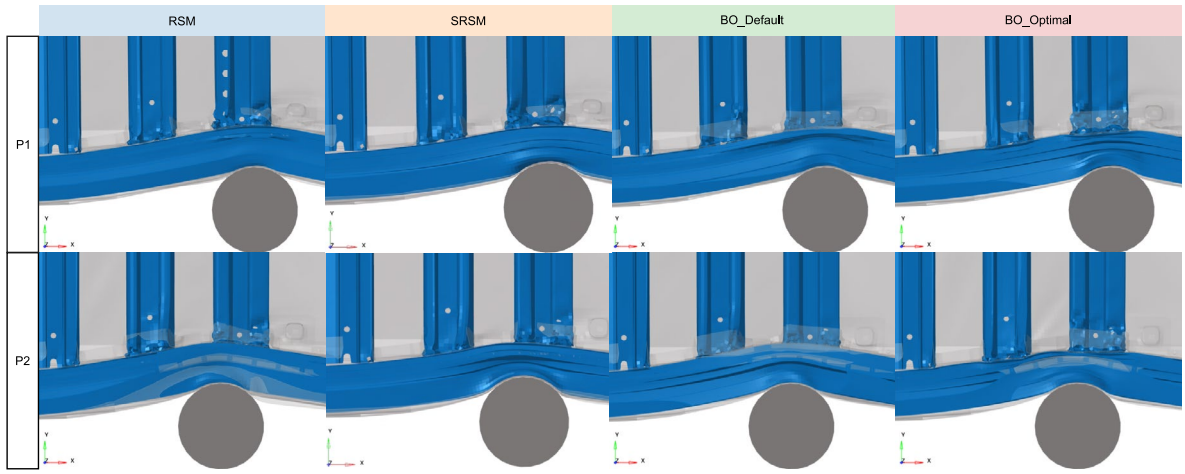
To evaluate the potential of our approach, two applications are considered, namely the BBOB suite and an automotive crash problem as a representative real-world expensive BBO problem. When applied to the BBOB functions in 20d, optimal configurations identified using our approach can most of the time outperform the default configuration in terms of optimization convergence speed quantified using the AUC metric. Moreover, optimal configurations found using our approach can compete against the SBS in some



(a) Comparison between the one-shot optimization strategy (*black*), RSM (*blue*), SRSM (*orange*), default BO configuration (*green*), and optimal BO configuration predicted using our approach (*red*) for the side crash problem with multi-pole impact (Equation 3). For RSM and SRSM, the best-found solution is presented. On the other hand, the best-so-far solution during re-sampling is shown for BO, starting with the best DoE sample. The median performance and standard deviation of three repetitions are reported. A smaller objective value and lower AUC is better.



(b) Comparison of vehicle designs for the side crash problem with multi-pole impact found by RSM (*blue*), SRSM (*orange*), default BO configuration (*green*), and optimal BO configuration identified using our approach (*red*), with the upper and lower bound of design space (thickness in mm) shown at the top and bottom row. The relative differences in mass (kg) and intrusions (mm) for P1 and P2 w.r.t. the vehicle design provided by RSM are shown. The optimization objective value is presented in the last column.



(c) Top view of the cross-sections of FE submodel, revealing the structural deformation during impact for vehicle designs found by RSM, SRSM, default BO, and optimal BO configuration identified using our approach for an impact at pole position P1 and P2. The same setting is applied for all simulation snapshots, e.g., deformation scaling, view position, and time frame.

Fig. 11 Overview of the optimization results using RSM, SRSM, default BO, and optimal BO configurations for the automotive side crash problem with a multi-pole impact

BBOB functions. Beyond that, our approach is flexible as it is compatible with different BBO algorithms, such as modular CMA-ES, modular DE, and BO, as well as different optimizers for HPO like TPE and SMAC.

Apart from the BBOB suite, our investigations are extended to an automotive side crash problem using FE simulations and two load cases, namely a single-pole and multi-pole impact. Generally, the performance of BO can be improved for both load cases, when optimally fine-tuned using our approach. Compared to SRSM, the advantage of using the optimal BO configuration identified is arguably minimal for the single-pole impact problem.

On the other hand, the optimal BO configuration can outperform SRSM in solving the multi-pole impact problem. In fact, an improvement in the vehicle design w.r.t. mass and intrusions can be identified using our approach for the multi-pole problem. Overall, we believe that our approach can be applied for solving automotive crash problems that require dealing with complex crash functions and a limited function evaluation budget, which are two crucial aspects in the automotive industry.

Based on our results, our approach has shown promising potential toward automated HPO in identifying optimal configurations for automotive crashworthiness optimization. Once the computational expense scales up, e.g., using a full vehicle FE model (Table 2), such an improvement in optimization efficiency could be substantial. Since the BBOB suite covers a wide range of optimization problem classes, we are confident that our approach can generalize to other BBO domains with similar optimization complexity. In this context, our approach is flexible for applications beyond automotive crash problems with modifications, such as ship design in the maritime industry (de Winter et al. 2019) and turbomachinery design in the aerospace industry (Pretsch et al. 2023).

In the meantime, several potential improvements in our approach have been noticed, which can be summarized as the following outlooks.

1. Currently, our approach is implemented for unconstrained single-objective optimization. To better handle real-world applications, we plan to extend our approach to constrained multi-objective optimization, and thereby, to increase its value for industrial applications.
2. An extension of our approach toward the so-called automated algorithm configuration (AAC) (Schede et al. 2022), and combined algorithm selection and hyperparameter optimization (CASH) (Thornton et al. 2013) can be another attractive research direction, where not only the hyperparameters, but also the choice of algorithm is part of the fine-tuning search space.
3. Another attractive research direction could be improving the generation of representative functions to better generalize our approach to a broader spectrum of expensive BBO problems. For future work, we plan to evaluate the performance of RGF generated (i) using our approach, (ii) starting from a GP prior, and (iii) using deep NN models (van Stein et al. 2023).
4. Moreover, we are motivated to develop a configurable BO framework that can offer a greater functional flexibility, similarly to the modular CMA-ES. In this regards, the BO hyperparameter search space considered in this work will be expanded, e.g., by combining different BO variants.

Acknowledgements The contribution of this paper was written as part of the joint project newAIDE under the consortium leadership of BMW AG with the partners Altair Engineering GmbH, divis intelligent solutions GmbH, MSC Software GmbH, Technical University of Munich, TWT GmbH. The project is supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag.

Author contributions Conceptualization: FL, NS, MF, PK, MG, and TB; Methodology: FL; Software: FL; Formal analysis and investigation: FL, NS, MF, PK, MG, and TB; Writing – original draft preparation: FL; Writing – review and editing: FL, NS, MF, MG, and TB; Supervision: TB and MG.

Funding Not applicable.

Data availibility Experimental results and figures are available in our repository https://zenodo.org/records/11350771. Refer to Sect. 5.1.

Conflict of interest The authors declare that they have no Conflict of interest.

Replication of results The implementation of the optimization approach is described in detail in Sect. 3 and the associated Python code can be obtained from the corresponding author on reasonable request. The automotive crash problems and FE submodel investigated in this work were developed internally by *BMW AG*, and thus are subject to strict confidentiality. While different outcomes are expected for other optimization problems, a comparable trend should be attainable.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

Antonov K, Raponi E, Wang H, et al (2022) High dimensional bayesian optimization with kernel principal component analysis. In: International Conference on Parallel Problem Solving from Nature, Springer, pp 118–131

Audet C, Hare W (2017) Derivative-free and blackbox optimization, vol 2. Springer, Berlin

Benjamins C, Jankovic A, Raponi E, et al (2022) Towards automated design of bayesian optimization via exploratory landscape analysis. arXiv preprint arXiv:2211.09678

Bergstra J, Bardenet R, Bengio Y, et al (2011) Algorithms for hyper-parameter optimization. In: Proceedings of the 24th International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, NIPS'11, p 2546-2554

Bergstra J, Yamins D, Cox D (2013) Making a science of model search: Hyperparameter optimization in hundreds of dimensions

for vision architectures. In: International conference on machine learning, PMLR, pp 115–123

Bossek J, Doerr C, Kerschke P, et al (2020) Evolving sampling strategies for one-shot optimization tasks. In: Bäck T, Preuss M, Deutz A, et al (eds) Parallel Problem Solving from Nature – PPSN XVI, vol 12269. Springer International Publishing, Cham, pp 111–124, https://doi.org/10.1007/978-3-030-58112-1_8

Czech C, Kaps A, Duddeck F (2022) Robust multi-fidelity optimization approach exploiting data-driven, non-linear model order reduction. In: 8th International Symposium on Reliability Engineering and Risk Management, pp 357–363

de Winter R, van Stein B, Dijkman M, et al (2019) Designing ships using constrained multi-objective efficient global optimization. In: Nicosia G, Pardalos P, Giuffrida G, et al (eds) Machine Learning, Optimization, and Data Science. Springer International Publishing, Cham, pp 191–203, https://doi.org/10.1007/978-3-030-13709-0_16

Doerr C, Wang H, Ye F, et al (2018) IOHprofiler: A Benchmarking and Profiling Tool for Iterative Optimization Heuristics. arXiv e-prints: 181005281 arXiv:1810.05281

Duddeck F (2008) Multidisciplinary optimization of car bodies. Struct Multidisc Optim 35(4):375–389. https://doi.org/10.1007/s00158-007-0130-6

Eriksson D, Pearce M, Gardner J et al (2019) Scalable global optimization via local bayesian optimization. Adv Neural Inform Proc Syst 32:5497–5508

Euro NCAP (2023) Far side occupant test & assessment protocol. https://cdn.euroncap.com/media/77295/euro-ncap-far-side-test-and-assessment-protocol-v24.pdf

Fang H, Rais-Rohani M, Liu Z et al (2005) A comparative study of metamodeling methods for multiobjective crashworthiness optimization. Comput Struct 83(25–26):2121–2136. https://doi.org/10.1016/j.compstruc.2005.02.025

Fang J, Sun G, Qiu N et al (2017) On design optimization for structural crashworthiness and its state of the art. Struct Multidisc Optim 55(3):1091–1119. https://doi.org/10.1007/s00158-016-1579-y

Gabry J, Simpson D, Vehtari A et al (2019) Visualization in Bayesian Workflow. J Royal Stat Soc 182(2):389–402. https://doi.org/10.1111/rssa.12378

Hamza K, Shalaby M (2014) A framework for parallelized efficient global optimization with application to vehicle crashworthiness optimization. Eng Optim 46(9):1200–1221. https://doi.org/10.1080/0305215X.2013.827672

Hansen N, Ostermeier A (1996) Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In: Proceedings of IEEE international conference on evolutionary computation, IEEE, pp 312–317

Hansen N, Finck S, Ros R, et al (2009) Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA, https://hal.inria.fr/inria-00362633

Hansen N, Auger A, Ros R, et al (2010) Comparing results of 31 algorithms from the black-box optimization benchmarking bbob-2009. In: Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, NY, USA, GECCO '10, p 1689-1696, https://doi.org/10.1145/1830761.1830790

Hansen N, Auger A, Ros R et al (2021) Coco: a platform for comparing continuous optimizers in a black-box setting. Optim Methods Softw 36(1):114–144. https://doi.org/10.1080/10556788.2020.1808977

Jones DR, Schonlau M, Welch WJ (1998) Efficient global optimization of expensive black-box functions. JGlobal Optim 13(4):455

Kaps A, Czech C, Duddeck F (2022) A hierarchical kriging approach for multi-fidelity optimization of automotive crashworthiness problems. Struct Multidisc Optim 65(4):114

Kerschke P, Trautmann H (2019) Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning. Evolut Comput 27(1):99–127. https://doi.org/10.1162/evco_a_00236

Kerschke P, Trautmann H (2019) Comprehensive Feature-Based Landscape Analysis of Continuous and Constrained Optimization Problems Using the R-Package Flacco, Springer International Publishing, Cham, pp 93–123. Stud Classif Data Anal Knowl Org. https://doi.org/10.1007/978-3-030-25147-5_7

Kerschke P, Preuss M, Wessing S, et al (2015) Detecting funnel structures by means of exploratory landscape analysis. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, NY, USA, GECCO '15, p 265-272,https://doi.org/10.1145/2739480.2754642

Kerschke P, Hoos H, Neumann F et al (2019) Automated algorithm selection: survey and perspectives. Evolut Comput 27(1):3–45. https://doi.org/10.1162/evco_a_00242

Kok S, Stander N (1999) Optimization of a sheet metal forming process using successive multipoint approximations. Struct Optim 18:277–295. https://doi.org/10.1007/BF01223312

Komer B, Bergstra J, Eliasmith C (2014) Hyperopt-sklearn: Automatic hyperparameter configuration for scikit-learn. In: SciPy, https://api.semanticscholar.org/CorpusID:6083252

Kurtaran H, Eskandarian A, Marzougui D et al (2002) Crashworthiness design optimization using successive response surface approximations. Comput Mech 29(4):409–421. https://doi.org/10.1007/s00466-002-0351-x

Li JY, Zhan ZH, Zhang J (2022) Evolutionary computation for expensive optimization: a survey. Machine Intel Res 19(1):3–23. https://doi.org/10.1007/s11633-022-1317-4

Lindauer M, Eggensperger K, Feurer M et al (2022) Smac3: a versatile bayesian optimization package for hyperparameter optimization. J Machine Learn Res 23(54):1–9

Lindauer M, Eggensperger K, Feurer M et al (2022) Smac3: A versatile bayesian optimization package for hyperparameter optimization. J Machine Learn Res 23(54):1–9

Livermore Software Technology Corporation (2019) Ls-dyna theory manual. https://ftp.lstc.com/anonymous/outgoing/jday/manuals/DRAFT_Theory.pdf

Long FX, van Stein B, Frenzel M, et al (2022) Learning the characteristics of engineering optimization problems with applications in automotive crash. In: Proceedings of the Genetic and Evolutionary Computation Conference. Association for Computing Machinery, New York, NY, USA, GECCO '22, p 1227-1236, https://doi.org/10.1145/3512290.3528712

Long FX, Vermetten D, Kononova AV, et al (2023a) Challenges of ELA-Guided Function Evolution Using Genetic Programming. In: Proceedings of the 15th International Joint Conference on Computational Intelligence - Volume 1: ECTA, INSTICC. SciTePress, pp 119–130, https://doi.org/10.5220/0012206200003595

Long FX, Vermetten D, van Stein B, et al (2023b) BBOB Instance Analysis: Landscape Properties and Algorithm Performance Across Problem Instances. In: Applications of Evolutionary Computation: 26th European Conference, EvoApplications 2023, Held as Part of EvoStar 2023, Brno, Czech Republic, April 12–14, 2023, Proceedings, Springer, pp 380–395, https://doi.org/10.1007/978-3-031-30229-9_25

Long FX, van Stein B, Frenzel M et al (2024) Generating cheap representative functions for expensive automotive crashworthiness optimization. ACM Trans Evol Learn Optim. https://doi.org/10.1145/3646554

Lunacek M, Whitley D (2006) The dispersion metric and the cma evolution strategy. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, NY, USA, GECCO '06, p 477-484, https://doi.org/10.1145/1143997.1144085

Malan KM (2021) A survey of advances in landscape analysis for optimisation. Algorithms 14(2):40. https://doi.org/10.3390/a14020040

Mersmann O, Preuss M, Trautmann H (2010) Benchmarking evolutionary algorithms: Towards exploratory landscape analysis. In: Schaefer R, Cotta C, Kołodziej J, et al (eds) Parallel Problem Solving from Nature, PPSN XI. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 73–82, https://doi.org/10.1007/978-3-642-15844-5_8

Mersmann O, Bischl B, Trautmann H, et al (2011) Exploratory landscape analysis. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, NY, USA, GECCO '11, p 829-836,https://doi.org/10.1145/2001576.2001690

Mockus J (1982) The bayesian approach to global optimization. System modeling and optimization. Springer, Berlin, pp 473–481

Muñoz MA, Kirley M, Halgamuge SK (2015) Exploratory landscape analysis of continuous space optimization problems using information content. IEEE Trans Evolut Comput 19(1):74–87. https://doi.org/10.1109/TEVC.2014.2302006

Muñoz MA, Sun Y, Kirley M et al (2015) Algorithm selection for black-box continuous optimization problems: a survey on methods and challenges. Inform Sci 317:224–245. https://doi.org/10.1016/j.ins.2015.05.010

Muñoz MA, Kirley M, Smith-Miles K (2022) Analyzing randomness effects on the reliability of exploratory landscape analysis. Nat Comput 21(2):131–154

de Nobel J, Vermetten D, Wang H, et al (2021) Tuning as a means of assessing the benefits of new ideas in interplay with existing algorithmic modules. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp 1375–1384

Pan F, Zhu P (2011) Design optimisation of vehicle roof structures: benefits of using multiple surrogates. Int J Crashworthiness 16(1):85–95

Pedregosa F, Varoquaux G, Gramfort A et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12(85):2825–2830

Prager RP, Trautmann H (2023a) Nullifying the inherent bias of non-invariant exploratory landscape analysis features. In: Applications of Evolutionary Computation: 26th European Conference, EvoApplications 2023, Held as Part of EvoStar 2023, Brno, Czech Republic, April 12-14, 2023, Proceedings. Springer-Verlag, Berlin, Heidelberg, p 411-425, https://doi.org/10.1007/978-3-031-30229-9_27

Prager RP, Trautmann H (2023) Pflacco: feature-based landscape analysis of continuous and constrained optimization problems in python. Evolut Comput. https://doi.org/10.1162/evco_a_00341

Pretsch L, Arsenyev I, Czech C et al (2023) Interdisciplinary design optimization of compressor blades combining low- and high-fidelity models. Struct Multidisc Optim. https://doi.org/10.1007/s00158-023-03516-w

Raponi E, Wang H, Bujny M, et al (2020) High dimensional bayesian optimization assisted by principal component analysis. In: Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part I 16, Springer, pp 169–183

Raponi E, Fiumarella D, Boria S et al (2021) Methodology for parameter identification on a thermoplastic composite crash absorber by the sequential response surface method and efficient global optimization. Composite Struct 278:114646. https://doi.org/10.1016/j.compstruct.2021.114646

Renau Q, Dreo J, Doerr C, et al (2019) Expressiveness and robustness of landscape features. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion. Association for Computing Machinery, New York, NY, USA, GECCO '19, p 2048-2051, https://doi.org/10.1145/3319619.3326913

Renau Q, Doerr C, Dreo J et al (2020) Exploratory landscape analysis is strongly sensitive to the sampling strategy. In: Bäck T, Preuss M, Deutz A et al (eds) Parallel Problem Solving from Nature - PPSN XVI. Springer International Publishing, Cham, pp 139–153. https://doi.org/10.1007/978-3-030-58115-2_10

Santoni ML, Raponi E, De Leone R, et al (2023) Comparison of high-dimensional bayesian optimization algorithms on bbob. arXiv preprint arXiv:2303.00890

Schede E, Brandt J, Tornede A et al (2022) A survey of methods for automated algorithm configuration. J Artif Intel Res 75:425–487

Sobol' IM (1967) On the distribution of points in a cube and the approximate evaluation of integrals. USSR Comput Math Math Phys 7(4):86–112. https://doi.org/10.1016/0041-5553(67)90144-9

Stander N, Craig K (2002a) Response surface and sensitivity-based optimization in ls-opt: A benchmark study. In: 7th International LS-DYNA Users Conference, Dearborn, MI

Stander N, Craig KJ (2002) On the robustness of a simple domain reduction scheme for simulation-based optimization. Eng Comput 19(4):431–450

Stander N, Roux W, Giger M, et al (2004) A comparison of metamodeling techniques for crashworthiness optimization. In: 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, p 4489

van Stein B, Wang H, Bäck T (2019) Automatic configuration of deep neural networks with parallel efficient global optimization. In: 2019 International Joint Conference on Neural Networks (IJCNN), IEEE, pp 1–7

van Stein B, Long FX, Frenzel M, et al (2023) Doe2vec: Deep-learning based features for exploratory landscape analysis. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. Association for Computing Machinery, New York, NY, USA, GECCO '23 Companion, p 515-518, https://doi.org/10.1145/3583133.3590609

van Stein N, Vermetten D, Kononova AV, et al (2024) Explainable benchmarking for iterative optimization heuristics. arXiv:2401.17842

Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11(4):341–359

Sun G, Tian Y, Wang R et al (2020) Parallelized multiobjective efficient global optimization algorithm and its applications. Struct Multidisc Optim 61(2):763–786. https://doi.org/10.1007/s00158-019-02417-1

Thomaser A, Kononova AV, Vogt ME, et al (2022) One-shot optimization for vehicle dynamics control systems: towards benchmarking and exploratory landscape analysis. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp 2036–2045

Thornton C, Hutter F, Hoos HH, et al (2013) Auto-weka: Combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Association for Computing Machinery, New York, NY, USA, KDD '13, p 847-855, https://doi.org/10.1145/2487575.2487629

Tian Y, Peng S, Zhang X et al (2020) A recommender system for metaheuristic algorithms for continuous optimization based on deep recurrent neural networks. IEEE Trans Artif Intel 1(1):5–18. https://doi.org/10.1109/TAI.2020.3022339

Vermetten D, Caraffini F, Kononova AV, et al (2023) Modular differential evolution. arXiv:2304.09524

Virtanen P, Gommers R, Oliphant TE et al (2020) SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods 17:261–272. https://doi.org/10.1038/s41592-019-0686-2

Škvorc U, Eftimov T, Korošec P (2021a) A Complementarity Analysis of the COCO Benchmark Problems and Artificially Generated Problems, Association for Computing Machinery, New York, NY, USA, p 215-216. https://doi.org/10.1145/3449726.3459585

Škvorc U, Eftimov T, Korošec P (2021b) The effect of sampling methods on the invariance to function transformations when using exploratory landscape analysis. In: 2021 IEEE Congress on Evolutionary Computation (CEC), pp 1139–1146, https://doi.org/10.1109/CEC45853.2021.9504739

Yildiz AR, Solanki KN (2012) Multi-objective optimization of vehicle crashworthiness using a new particle swarm based approach. Int J Adv Manuf Technol 59:367–376

Zhao M, Li J (2018) Tuning the hyper-parameters of cma-es with tree-structured parzen estimators. In: 2018 Tenth International Conference on Advanced Computational Intelligence (ICACI), pp 613–618,https://doi.org/10.1109/ICACI.2018.8377530

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.