Generating cheap representative functions for expensive automotive crashworthiness optimization

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

Generating Cheap Representative Functions for Expensive Automotive Crashworthiness Optimization

FU XING LONG, BMW Group, Munich, Germany
BAS VAN STEIN, LIACS, Leiden University, Leiden, The Netherlands
MORITZ FRENZEL, BMW Group, Munich, Germany
PETER KRAUSE, divis intelligent solutions GmbH, Dortmund, Germany
MARKUS GITTERLE, University of Applied Sciences Munich, Munich, Germany
THOMAS BÄCK, LIACS, Leiden University, Leiden, The Netherlands

Solving real-world engineering optimization problems, such as automotive crashworthiness optimization, is extremely challenging, because the problem characteristics are oftentimes not well understood. Furthermore, typical hyperparameter optimization (HPO) approaches that require a large function evaluation budget are computationally hindered, if the function evaluation is expensive, for example, requires finite element (FE) simulation runs. In this article, we propose an approach to characterize real-world expensive black-box optimization problems using the exploratory landscape analysis (ELA). Based on these landscape characteristics, we can identify test functions that are fast-to-evaluate and representative for HPO purposes. Focusing on 20 problem instances from automotive crashworthiness optimization, our results reveal that these 20 crashworthiness problems exhibit landscape features different from classical optimization benchmark test suites, such as the widely-used black-box optimization benchmarking (BBOB) problem set. In fact, these 20 problem instances belong to problem classes that are distinct from the BBOB test functions based on the clustering results. Further analysis indicates that, as far as the ELA features concern, they are most similar to problem classes of tree-based test functions. By analyzing the performance of two optimization algorithms with different hyperparameters, namely the covariance matrix adaptation evolutionary strategy (CMA-ES) and Bayesian optimization (BO), we show that the tree-based test functions are indeed representative in terms of predicting the algorithm performances. Following this, such scalable and fast-to-evaluate tree-based test functions have promising potential for automated design of an optimization algorithm for specific real-world problem classes.

CCS Concepts: • **Computing methodologies** → **Continuous space search**; **Uncertainty quantification**; • **Mathematics of computing** → **Continuous optimization**; • **Applied computing** → **Engineering;**

Additional Key Words and Phrases: Automotive crashworthiness, black-box optimization, single-objective, exploratory landscape analysis, representative functions

The contribution of this article 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.
Authors' addresses: F. X. Long and M. Frenzel, BMW Group, Knorrstraße 147, Munich, Bavaria, DE-80788, Germany; e-mails: fu-xing.long@bmw.de, moritz.frenzel@bmw.de; B. van Stein and T. Bäck, LIACS, Leiden University, Niels Bohrweg 1, Leiden, South Holland, NL-2333, The Netherlands; e-mail: b.van.stein@liacs.leidenuniv.nl, t.h.w.baeck@liacs.leidenuniv.nl; P. Krause, Divis Intelligent Solutions GmbH, Joseph-von-Fraunhofer-Straße 20, Dortmund, North Rhine-Westphalia, DE-44227, Germany; e-mail: krause@divis-gmbh.de; M. Gitterle, University of Applied Sciences Munich, Dachauer Straße 98b, Munich, Bavaria, DE-80335, Germany; e-mail: markus.gitterle@hm.edu.

ACM 2688-3007/2024/06-ART9
https://doi.org/10.1145/3646554

ACM Reference Format:
Fu Xing Long, Bas van Stein, Moritz Frenzel, Peter Krause, Markus Gitterle, and Thomas Bäck. 2024. Generating Cheap Representative Functions for Expensive Automotive Crashworthiness Optimization. *ACM Trans. Evol. Learn. Optim.* 4, 2, Article 9 (June 2024), 26 pages. https://doi.org/10.1145/3646554

1 INTRODUCTION

When solving **black-box optimization** (**BBO**) problems, identifying and selecting the most time efficient- and resource-efficient algorithm is crucial to ensure an optimal optimization performance, which is also known as the **algorithm selection problem** (**ASP**) [Rice 1976]. In evolutionary computation, recent works focus on landscape-aware automated algorithm selection based on **machine learning** (**ML**) approaches, such as in [Bischl et al. 2012; Dréo et al. 2019; Jankovic and Doerr 2020; Jankovic et al. 2021; Kerschke et al. 2019; Kerschke and Trautmann 2019a; Pikalov and Mironovich 2021]. In these approaches, the so-called landscape features, which quantify different landscape characteristics of a given problem instance, are employed to predict the performance of an optimization algorithm on the problem instance, typically using supervised ML models. In other words, the performance of an optimization algorithm on an unseen problem instance can be roughly estimated, once the problem landscape characteristics have been identified. Following this, the best possible optimization algorithm for a problem instance can be identified based on its landscape properties. Furthermore, these landscape features can provide additional insights for explaining the effectiveness of an algorithm across different problem instances [Simoncini et al. 2018]. Complete reviews of this topic can be found in [Kerschke et al. 2019; Malan 2021; Muñoz et al. 2015b].

To the best of our knowledge, previous works on ASP were mainly based on academic benchmark functions, such as the well-known **black-box optimization benchmarking** (**BBOB**) test functions. On the other hand, little work has been done to tackle ASP from real-world expensive BBO domain, such as crashworthiness optimization in automotive industry. The goal of automotive crashworthiness optimization is to ensure that the structural design of a vehicle can provide sufficient protection to passengers in the event of a crash, while fulfilling other requirements at the same time, such as durability and weight [Duddeck 2008]. As vehicle design is getting ever more sophisticated nowadays, crashworthiness optimization is notoriously challenging and tedious. Due to the fact that the problem properties or landscape characteristics of automotive crashworthiness optimization problems are not well understood and the function evaluation is computationally expensive, for example, through **finite element** (**FE**) simulations, typical ASP approaches are not directly applicable for these problems.

Our long-term vision is to develop an automated optimization pipeline for automotive crashworthiness optimization that can assist engineers in identifying the best possible optimization algorithm for their applications, effectively accelerating the progress of vehicle development projects. Not only limited to automotive crashworthiness optimization, our pipeline is designed with the aim that it can be extended and applied in other expensive BBO domains with modifications. An overview of the workflow of our pipeline is presented in Figure 1. In real-world BBO problems like automotive crashworthiness optimization, the main challenge of identifying the best optimization algorithm is the computationally expensive FE simulations, which severely limit the evaluation budget affordable for optimization. To overcome this hurdle, our idea is to first identify appropriate test functions that have similar problem characteristics as a BBO problem and consider them as *representative functions* for the BBO problem. Since these representative functions belong to the same problem class as the BBO problem, the performance of an optimization algorithm should be similar on both problems. Following these, the representative functions can be exploited to predict

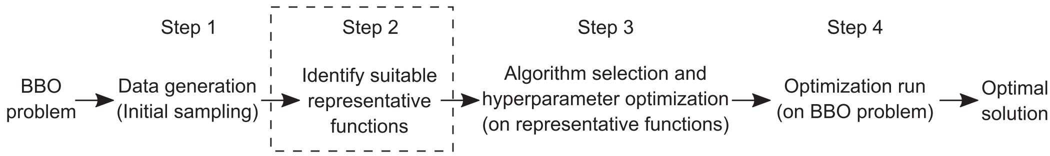


Fig. 1. Our automated optimization pipeline starts with a data set of a BBO problem as input in Step 1, for example, a crash problem. Generally, the idea is to identify appropriate representative functions that have similar landscape characteristics to this BBO problem instance in Step 2, which can be exploited to predict optimal optimization algorithms in Step 3. Lastly, an optimal or the best optimization algorithm will be applied on the BBO problem instance in Step 4. This work focuses on Step 2 in the pipeline, as outlined.

the estimated algorithm performances on the actual BBO problem. More importantly, the representative functions are fast-to-evaluate in comparison to FE simulations, thus allowing a more flexible function evaluation budget for algorithm selection and **hyperparameter optimization** (**HPO**) purposes.

In this article, we propose an approach for characterizing and identifying appropriate representative functions for expensive BBO problems based on landscape characteristics using the **exploratory landscape analysis** (**ELA**), refer to Step 2 in Figure 1. In this context, we consider several automotive crashworthiness optimization problem instances as our test cases. Precisely, we are addressing the following research questions:

(1) Within the feature space defined by the ELA features, how are the distributions of automotive crashworthiness optimization and BBOB functions situated w.r.t. each other?
(2) If none of the BBOB functions is sufficiently representative, how to augment the BBOB functions with other test functions, in order to support the finding of representative functions?
(3) Can we predict the estimated optimization algorithm performances on problem instances based on their corresponding representative functions?

The remainder of this article is structured as follows: Section 2 briefly summarizes the related work. This is followed by the description of our methodology in Section 3 and an overview of the experimental setup in Section 4. Our results concerning the landscape characteristics of automotive crashworthiness optimization are discussed in Section 5, representative functions for automotive crash in Section 6, and prediction of algorithm performances in Section 7. Lastly, conclusions and future works are presented in Section 8.

2 RELATED WORK

As an effort in filling the gap in optimization problem classes not being covered by optimization benchmark suites, for example, the BBOB suite, various approaches to generate test functions of new problem classes have been previously introduced. For instance, an approach based on affine combinations of two BBOB functions via interpolation was proposed in [Dietrich and Mersmann 2022], which was later extended to affine combinations of many BBOB functions in [Vermetten et al. 2023b, a]. In [Tian et al. 2020a], a function generator that can randomly generate tree-based test functions of different complexity was introduced, which belong to problem classes different from the BBOB functions, as shown in [Škvorc et al. 2021a]. Based on that work, genetic programming was recently employed to evolve functions with desired landscape characteristics in [Long et al. 2023a]. Another interesting research direction is the so-called instance space analysis in [Smith-Miles and Muñoz 2023].

This work is an extension of the previous work [Long et al. 2022]. To the best of our knowledge, (i) investigating the problem class of real-world automotive crashworthiness optimization, and

(ii) identifying representative functions with similar landscape characteristics for algorithm selection and HPO purposes are still lacking. While considering data-driven surrogate models based on some problem samples as representative functions for HPO purposes could be another more straightforward alternative, our approach has the advantage that the algorithm hyperparameters are optimized to the problem class, rather than to that particular problem only. That is, the optimal hyperparameters identified will be appropriate for other problem instances from the same problem class, without requiring additional experiments. Beyond that, we are not interested in surrogates or proxies that have the exact landscape, or the optimum in the same location, but aim to establish a preferably large set of test functions with similar landscape characteristics for HPO purposes.

2.1 Automotive Crashworthiness Optimization

In automotive crashworthiness optimization problems, or simply automotive crash problems in the remainder of this article, the fitness function can be typically defined as minimization of the magnitude of intrusion or maximization of the energy absorption of certain components [Fang et al. 2017]. They belong to the category of BBO problems, where derivative information is lacking. Typically, the classical one-shot optimization is applied for automotive crash problems, where the power of parallel computing can be exploited for parallel FE simulation runs to generate a large set of **design of experiments** (**DoE**) samples [Bossek et al. 2020; Bousquet et al. 2017]. Basically, one-shot optimization can be divided into two approaches: (i) simply consider the best sample from the DoE as solution, without evaluating new samples, and (ii) train a surrogate model based on the DoE samples to predict better solutions. In the second approach, one or more solution(s) will be evaluated using FE simulations in a second step. While many previous works focused on the selection of surrogate models to approximate the true automotive crash problems, such as in [Fang et al. 2005; Forsberg and Nilsson 2006; Jansson et al. 2003; Kiani et al. 2013; Kurtaran et al. 2002; Liao et al. 2008; Shi et al. 2012], the surrogate-based one-shot optimization method is still inefficient, because the fitting quality of surrogate models is oftentimes rather poor due to the strongly nonlinear and discontinuous characteristics of automotive crash problems. Multi-fidelity optimization and model order reduction are some other active research areas for automotive crashworthiness optimization [Czech et al. 2022; Kaps et al. 2022].

Another promising research direction is the application of the **efficient global optimization** (**EGO**) [Jones et al. 1998] on automotive crash problems, which was based on the **Bayesian optimization** (**BO**) [Mockus 1982]. In fact, solving automotive crash problems with the EGO has shown promising potential in [Hamza and Shalaby 2014; Sun et al. 2020]. On the other hand, while state-of-the-art derivative-free optimization algorithms, such as the **covariance matrix adaptation evolution strategy** (**CMA-ES**) [Hansen and Ostermeier 1996], are powerful to solve BBO problems, they are less applicable for automotive crash problems, since they usually demand a large function evaluation budget for convergence. As the performance of iterative optimization algorithms, like EGO and CMA-ES, is sensitive to their hyperparameters, HPO is essential, yet challenging without expert knowledge.

2.2 Black-Box Optimization Benchmarking

Over the years, optimization benchmark suites have been introduced to facilitate the performance evaluation of different optimization heuristics algorithms. For instance, the original BBOB suite in [Hansen et al. 2009] is one of the most well-known set of benchmark functions, such as in [Hansen et al. 2010], consisting of altogether 24 single-objective, noiseless, and continuous functions. We refer to this suite as *the* BBOB throughout this article. Recently, the BBOB suite has been integrated in the **comparing continuous optimizers (COCO)** platform [Hansen et al. 2021] and **iterative optimization heuristics profiler** (**IOHProfiler**) tool [Doerr et al. 2018] for benchmarking purposes.

While the BBOB suite was initially designed for unconstrained optimization, they are commonly considered within the search domain of $[-5, 5]^d$ with their global optimum located within $[-4, 4]^d$, where d represents dimensionality. Apart from the fact that they can be scaled up to arbitrary dimensionality, different variants or problem instances of the BBOB functions can be easily generated through a transformation of the search domain and objective values, which is internally controlled by a unique identifier, or also known as IID. Essentially, problem instances of the same function belong to the same problem class. Detailed analysis of the BBOB instances is available in [Long et al. 2023b].

2.3 Exploratory Landscape Analysis

One way to characterize the complexity of continuous optimization problems is through the so-called high-level properties, such as multi-modality, global structure, and separability [Mersmann et al. 2010]. To facilitate an automated characterization of problem instances, the ELA has been proposed in [Mersmann et al. 2011] to numerically quantify six classes of low-level landscape properties, namely y-distribution, level set, meta-model, local search, curvature, and convexity. Later, additional feature classes were introduced to complement these six classical ELA feature classes, consisting of dispersion, **nearest better clustering** (**NBC**), **principal component analysis** (**PCA**), linear model, and **information content of fitness sequences** (**ICoFiS**) [Kerschke et al. 2015; Kerschke and Trautmann 2019b; Lunacek and Whitley 2006; Muñoz et al. 2015a]. In fact, the results in [Renau et al. 2021] revealed that the ELA features are sufficiently informative for a classification task of the BBOB functions.

Basically, a DoE of some samples $\mathcal{X} = \{x_1, \cdots, x_n\}$ evaluated on an objective function f, that is, $f : \mathbb{R}^d \to \mathbb{R}$, is required as input to compute the ELA features, where $x_i \in \mathbb{R}^d$ and n represents the number of samples. While it has been shown that the ELA features are highly sensitive to the DoE sample size and sampling strategy in [Muñoz et al. 2022; Renau et al. 2019; Škvorc et al. 2021b], these topics are beyond the focus of this work.

Apart from handling ASP tasks, the ELA has been applied to analyze the landscape properties of vehicle dynamics control systems [Thomaser et al. 2022], to understand the optimization landscape of neural architecture search tasks [van Stein et al. 2020], to analyze the problem space of different benchmark problem sets [Škvorc et al. 2020], and to investigate whether the BBOB test function set can represent hyperparameter tuning problems [Doerr et al. 2019]. Recently, [Benjamins et al. 2022] investigated the selection of optimal hyperparameter, precisely the acquisition function, for the BO algorithm based on the ELA features. As far as we are aware, no previous work is related to analyzing the landscape characteristics of automotive crash problems using the ELA approach.

3 METHODOLOGY

To identify appropriate representative functions for a BBO problem, for example, an automotive crash problem in this work, we consider two crucial aspects in our pipeline, namely (i) characterization of the landscape properties of the given problem using the ELA, and (ii) searching for similar test functions in terms of their landscape properties. In other words, the ELA features of the BBO problem are compared with those of some test functions, such as the BBOB functions. Subsequently, test functions with a small difference in the ELA features are considered as representative functions for the BBO problem. The proposed workflow is visualized in Figure 2 and explained in detail in the following.

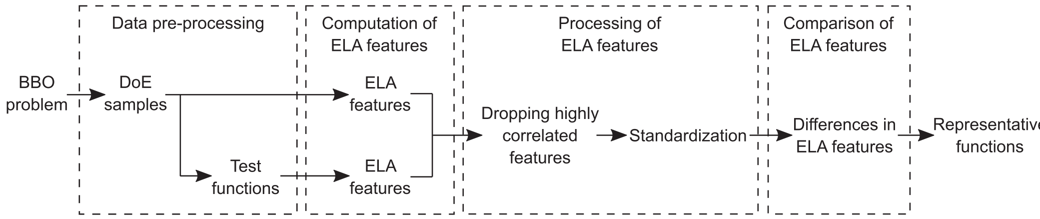


Fig. 2. The detailed workflow of Step 2 in Figure 1 that requires some DoE samples as input, and consists of four sections as marked with boxes. Essentially, we first characterize a BBO problem instance by computing its ELA features. Next, we compare them with those of some test functions, for example, the BBOB functions, to identify appropriate representative functions based on the differences in ELA features.

Data pre-processing. Firstly, the input DoE data of a BBO problem are pre-processed, where duplicated samples or samples with incomplete data, for example, missing results due to interrupted FE simulation runs, are discarded. To facilitate the comparison of ELA features between the BBO problem and test functions, the search space of the BBO problem is re-scaled using Equation (1).

$$x_{new} = \frac{x_{orig} - a_{min}}{a_{max} - a_{min}} \cdot (b_{max} - b_{min}) + b_{min}\,, \tag{1}$$

where x_{orig} and x_{new} are the design variables before and after re-scaling, a_{min} and a_{max} are the original minimum and maximum scale range, and b_{min} and b_{max} are the minimum and maximum scale range after re-scaling. Here, we consider the domain $[-5, 5]^d$ commonly used for the BBOB functions. Lastly, the same DoE samples, and thus the same dimensionality, are utilized to compute the objective values of the test functions.

Computation of ELA features. Among the more than 300 available ELA features, we (i) consider only the “cheap” ELA features that can be computed without additional re-sampling, (ii) neglect those that concern only the DoE samples $\mathcal{X}$, for example, some of the PCA features, and (iii) ignore those regarding the computational costs, which are not informative about the problem landscape. For expensive BBO problems, since the analytical form is oftentimes not known and function evaluation is costly, consideration of ELA features that require re-sampling is particularly limited. Altogether 49 ELA features from seven feature classes are separately computed on the BBO problem and test functions, as summarized in Table 1. Additionally, we normalize the objective values using min-max scaling before the ELA feature computation to remove inherent bias, as suggested in [Prager and Trautmann 2023a].

For the ELA feature computation, we integrate the `pflacco` package [Prager 2022; Prager and Trautmann 2023b] into our pipeline, which was implemented based on the `flacco` package [Kerschke and Trautmann 2019c]. In cases where a feature computation fails, for example, when the sample size is too small for computing the level set features, it will be skipped. Consequently, less ELA features will be computed for such a problem instance.

Processing of ELA features. Due to the fact that many of the ELA features are redundant, as shown in [Renau et al. 2019; Škvorc et al. 2020], we consider only a subset of all the computed ELA features, where ELA features with a constant value across all functions are automatically neglected. Precisely, we remove highly correlated ELA features based on the Pearson’s correlation coefficient, where for each highly correlated feature pair, the feature that has a higher mean correlation with the remaining features is removed. Furthermore, we standardize the remaining ELA features by removing mean and scaling to unit variance. Both steps are first performed on the ELA features

Table 1. Brief Descriptions of the 49 ELA Features from Seven ELA Feature Classes Considered in this Work, with the Respective Labels for Feature Classes and ELA Features

Feature class	Description	ELA feature

y-distribution (`ela_distr.*`)	Distribution of function values. 3 features	`skewness`
		`kurtosis`
		`number_of_peaks`

Level set (`ela_level.*`)	Measure the performance of different classification methods based on function value thresholds. 9 features	`mmce_lda_{10,25,50}`
		`mmce_qda_{10,25,50}`
		`lda_qda_{10,25,50}`

Meta-model (`ela_meta.*`)	Fitting quality of linear and quadratic models with and without interactions. 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 (`disp.*`)	Comparison of dispersion between initial sample points and subset of points based on function value thresholds. 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 (`nbc.*`)	Comparison of distance between all sample points towards nearest points and nearest points with better function value. 5 features	`nn_nb.{sd_ratio,mean_ratio,cor}`
		`dist_ratio.coeff_var`
		`nb._fitness.cor`

PCA (`pca.*`)	Information based on PCA on initial sample points. 2 features	`expl_var_PC1.{cov_init,cor_init}`

ICoFiS (`ic.*`)	Measure of smoothness, ruggedness and neutrality of the landscape through random walk. 5 features	`h.max`
		`eps.{s,max,ratio}`
		`m0`

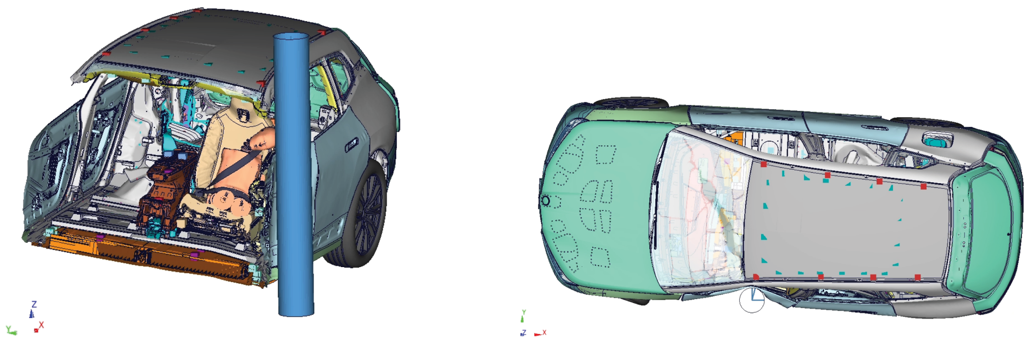
of BBOB functions, and then applied on those of the BBO problem, using the same subset of ELA features and standardizing with the same mean and variance.

Comparison of ELA features. To quantify the similarity between the BBO problem and test functions, we compute the differences in their ELA features based on the Euclidean distance metric. Following this, the test function that has the smallest distance, that is, the smallest difference in ELA features, is considered as the representative function for the BBO problem.

4 EXPERIMENTAL SETUP

4.1 Automotive Side Crash

In this work, we analyze different automotive crash scenarios, consisting of side crash, rear crash, roof crash, and front crash. Particularly, we focus on automotive side crash against a pole as the representative crash scenario, where the battery cells installed in an electric car must be additionally protected from crash impact, as shown in Figure 3. All the FE simulation data were generated during several previous development projects by BMW, a German premium automobile manufacturer, using the commercial solver LS-DYNA [Livermore Software Technology Corporation 2019].



(a) Cross-section of the deformed FE model in a side crash. To protect passengers and battery cells, the crash impact energy must be sufficiently absorbed through plastic deformation of different components, such as the rocker panel.

(b) Depending on the investigation purposes, the side pole can be positioned at different locations alongside the car body.

Fig. 3. An example of FE model developed for investigating automotive side crash against a pole.

As summarized in Table 2, we consider altogether 20 automotive crash problems, where the **modified extensible lattice sequence** (**MELS**) sampling method available in the commercial tool `HyperStudy` [Altair Engineering Inc. 2022] was utilized. Basically, the MELS is a sequential lattice space-filling DoE approach developed based on the Sobol' sequences [Sobol' 1967]. While the DoE sample size could be too small for reliable ELA features computation, particularly in the problem instance Crash_1, which is actually common for automotive crash problems due to the expensive FE simulations, adding more samples to the DoE is ruled out in our research, since we have no access to these FE simulations. Within the same crash scenario, the problem instances were mainly different in terms of vehicle models and load cases, for example, different pole positions for side crash. Furthermore, the design variables were the thicknesses of different vehicle components to be optimized, for example, the thicknesses of rocker panels for side crash scenario.

During these development projects, the quality of a particular component design was evaluated using the following five objectives, which were measured and quantified as scalar FE outputs. We also use the term *crash functions* for these objectives.

(1) Mass (M): Weight of components.
(2) Maximum force (F_{max}): Maximum impact force during crash.
(3) Intrusion ($Intr$): Magnitude of inward structural deformation.
(4) Energy absorption (EA): The amount of kinetic energy absorbed during crash.
(5) Rotation (Rot): Rotational deformation of components during crash. This metric was introduced to measure the average vertical deformation of FE nodes between inner and outer side of components.

While the mass objective was mainly concerning the manufacturing costs and car weight, the remaining objectives shed light on the structural crashworthiness of a component design. Nonetheless, depending on the purposes of each development project, not all five objectives were always considered and, therefore, not all of them are available for our study. In fact, out of the maximum of 100 potentially available ones, only a total of 48 crash functions are available. Within the scope of this work, we separately analyze each crash function as unconstrained single-objective optimization problem.

Table 2. Summary of 20 Automotive Crash Problem Instances Analyzed in this Work, Consisting of Four Different Crash Scenarios

Problem instance	Scenario	Design variables	Sample size
Crash_1	Side crash	22	59
Crash_2	Side crash	22	309
Crash_3	Side crash	22	309
Crash_4	Side crash	16	150
Crash_5	Side crash	16	102
Crash_6	Side crash	16	132
Crash_7	Side crash	20	329
Crash_8	Side crash	20	330
Crash_9	Side crash	20	333
Crash_10	Side crash	18	530
Crash_11	Side crash	13	150
Crash_12	Side crash	14	99
Crash_13	Rear crash	12	180
Crash_14	Rear crash	12	259
Crash_15	Roof crash	19	100
Crash_16	Roof crash	17	107
Crash_17	Front crash	22	487
Crash_18	Front crash	20	246
Crash_19	Front crash	8	248
Crash_20	Front crash	8	246

Using the pipeline introduced in Section 3, we independently compute the ELA features of each crash function. To minimize the effects of random sampling in the ELA, we consider the mean ELA feature values computed based on a bootstrapping strategy. Precisely, the ELA features are repeatedly computed for 20 times based on a subset of the DoE data, consisting of 80% of the DoE samples that are selected using different random seeds in each repetition, and then averaged. Furthermore, we consider the BBOB functions as test functions, where we additionally average the ELA feature values across the first 20 BBOB instances. In this work, we consider an ELA feature pair with a Pearson's correlation coefficient greater than 0.95 as highly correlated.

4.2 Performance of Optimization Algorithms

To evaluate the representativeness of test functions for a BBO problem, we compare the ranking of optimization algorithms with identical hyperparameter settings based on their performances on both problems. Since testing on the crash functions is extremely time-consuming and the fact that we have no opportunity to conduct additional FE simulations, our investigation focuses on the 24 BBOB functions of the first instance. In brief, for each of the BBOB functions of 20 dimensions, or simply $20d$, we first identify representative functions based on their landscape similarity using a similar experimental setup, with a DoE of 400 samples generated using the Sobol' sequences based on the results in [Renau et al. 2020] and bootstrapping with 30 repetitions.

For the optimization, we consider two state-of-the-art algorithms for BBO, namely the CMA-ES [de Nobel et al. 2021] and BO algorithm [Nogueira 2014], and the set of algorithm hyperparameters summarized in Table 3. When using the BO algorithm, the same DoE samples for the ELA features computation are utilized to build the **Gaussian process** (**GP**) model. Using the exhaustive grid search approach, we evaluate a total of 972 configurations, that is, different hyperparameter

Table 3. Summary of Hyperparameters Considered for the CMA-ES and BO Algorithm

Algorithm	Hyperparameter	Value
CMA-ES	Number of offspring λ	{ 10, 20 }
	Number of parents μ	{ 3, 5 }
	Initial standard deviation σ_0	{ 0.1, 0.3, 0.5 }
	Learning rate step size control C_σ	{ 0.1, 0.5, 1.0 }
	Learning rate covariance matrix adaptation C_c	{ 0.1, 0.5, 1.0 }
	Learning rate rank-μ update C_μ	{ 0.1, 0.5, 1.0 }
	Learning rate rank-one update C_1	{ 0.1, 0.5, 1.0 }
BO	DoE sample size	{ 50, 150, 250 }
	Acquisition function	{ Expected improvement (EI), probability of improvement (PI), upper confidence bound (UCB) }

settings, for CMA-ES and nine for BO separately on both the BBOB functions and their representative functions. Each configuration is allocated a fixed evaluation budget of 2, 000 evaluations for CMA-ES and 300 for BO, and its median performance over 30 repetitions using different random seeds is reported. Since the time-complexity of training a GP model rapidly increases with the number of samples, we consider a smaller budget for the BO algorithm. While HPO could potentially further improve the optimization performances, finding the best hyperparameter setting is not the focus of this work. In fact, we are interested in evaluating the potential of predicting algorithm performances based on representative functions.

To evaluate the performance of an algorithm configuration, we consider two metrics, namely the quality of optimization solution and **area under the curve** (**AUC**) of optimization runs, refer to Figure 12. The AUC metric can be useful in indicating the optimization convergence speed, which is crucial in real-world scenarios, as finding a good solution within a shorter time is oftentimes preferable than finding the best solution. Furthermore, the AUC metric is also informative, when multiple algorithms are equally competitive in delivering the same solution. Since we consider minimization problems in this work, an algorithm is considered better, if it can provide solutions with smaller objective values and/or smaller AUC values.

5 LANDSCAPE CHARACTERISTICS OF AUTOMOTIVE CRASHWORTHINESS OPTIMIZATION

In the first step, we begin our investigation by analyzing the problem class of automotive crash problems through comparison of their landscape characteristics to those of BBOB functions using the agglomerative hierarchical clustering approach [Murtagh and Contreras 2012]. In this approach, problems are clustered together in a bottom up fashion, starting with each problem as its own cluster and progressively merging clusters together until one large cluster is left, consisting of all problems. In this context, we consider the Ward's method [Ward Jr 1963] as linkage criterion for the cluster merging strategy, that is, by minimizing the within-cluster variance. In other words, clusters are selected for merging based on the smallest possible increase in the within-cluster sum of squared error after merging, which is proportional to the Euclidean distance. Following this, the problem class of automotive crash problems can be easily identified based on the BBOB functions within the same cluster.

The clustering results are visualized in Figure 4. In this work, we consider the average pairwise distance between different BBOB functions as our reference distance in determining to which extent a crash function is similar to its neighbouring BBOB functions. Generally, we observe that

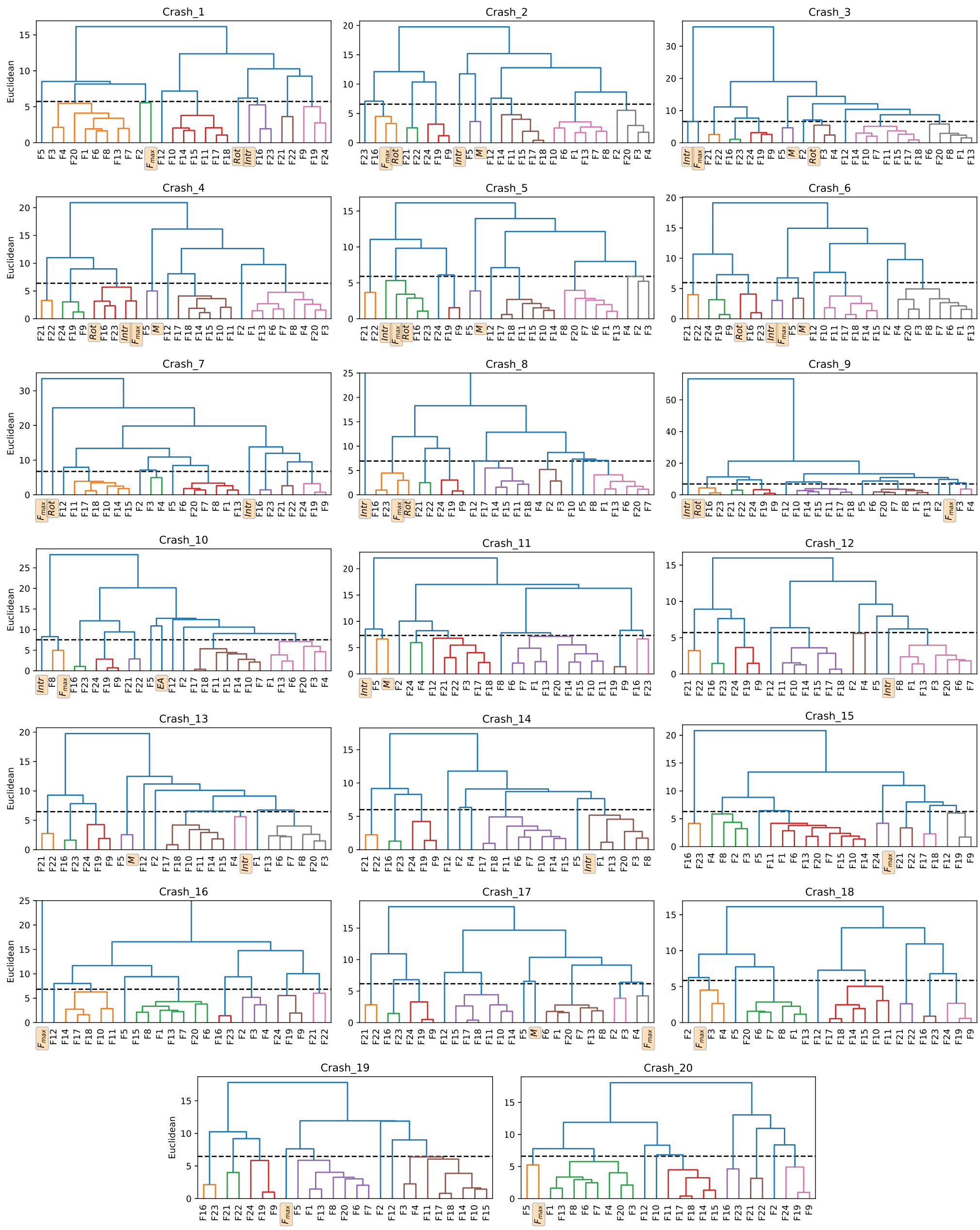


Fig. 4. The clustering patterns of crash functions and 24 BBOB functions for all 20 automotive crash problems, with the labels of crash functions highlighted in orange color. The reference Euclidean distance is marked with a dashed line, and clusters below it are assigned with different colors. The intrusion function in Crash_8 has a Euclidean distance of around 700 to the main cluster, and the maximum force function in Crash_16 has a distance of around 300, which are cut-off due to visualization purposes.

many of the crash functions are rather separated from the BBOB functions, especially the intrusion function in Crash_8 and Crash_9, and the maximum force function in Crash_16. The extremely large Euclidean distance of the intrusion function in Crash_8 is primarily due to the standardized *pca.expl_var_PC1.cov_init* feature. This indicates that our standardization approach for ELA features using the mean and variance from BBOB functions might not be appropriate for all crash functions. The fact that some crash functions, such as rotation function in Crash_6, are clustered in the same group as F16 (Weierstrass) and F23 (Katsuura) suggests that they could have similar landscape properties, for example, highly rugged and repetitive landscape. Noticeably, all mass functions are clustered in the same group as F5 (linear slope) as expected, since mass is linearly dependent on the thicknesses of vehicle components. Based on the clustering patterns, we observe that many automotive crash problems are different from the BBOB functions in terms of landscape characteristics and belong to other problem classes. Consequently, the BBOB functions might be not sufficiently representative for automotive crash problems. This will be discussed further in Section 6.

To have a better understanding of the clustering results, we delve into examining each individual ELA feature, using Crash_2 as an example, because it has the highest dimensionality and sufficiently large DoE sample size. By visually inspecting Figure 5, we notice that several ELA features show remarkable differences in feature values between the crash and BBOB functions, for instance, *ela_distr.kurtosis* and *ela_level.mmce_qda_10*. For an unbiased analysis, we consider the two-sample **Kolmogorov-Smirnov** (**KS**) test [Massey Jr 1951] available in `scipy` [Virtanen et al. 2020] to compare the ELA feature distributions, with the null hypothesis that the distribution of ELA features between the crash and BBOB functions is similar. The fact that the null hypothesis is rejected for some ELA features with a confidence level of 95% indicates that the distribution of these ELA features are indeed not similar between the crash and BBOB functions, which might explain the separation observed in the previous clustering patterns.

The comparison of ELA feature distributions between the crash and BBOB functions for all 20 automotive crash problems is presented in Figure 6. While no obvious ELA feature has completely different distribution between the crash and BBOB functions in all 20 problem instances, we observe that the null hypothesis is rejected in many of the dispersion features. This indicates that the crash and BBOB functions might have different degree of dispersion, which quantifies the size of search space region with better solutions.

Moreover, using the **t-distributed stochastic neighbor embedding** (**t-SNE**) approach [van der Maaten and Hinton 2008] available in `sklearn` [Pedregosa et al. 2011]), we project the high-dimensional ELA feature space on a $2d$ visualization for all crash and BBOB functions, as shown in Figure 7. Similar to the observations in our results earlier, many of the crash functions are indeed separated from the BBOB functions, indicating that these functions belong to different problem classes. Surprisingly, rather than forming a cluster, the crash functions are spread across the ELA feature space, forming their own problem classes, even for crash functions of the same type, for example, intrusion. Accordingly, this insight suggests that representative functions must be separately identified for each problem class of crash functions. Nonetheless, we are aware that the experimental setup of this work might be not optimal to reach a decisive conclusion, since ELA features are sensitive to factors, such as DoE sample size and problem dimensionality, and further in-depth investigations are necessary.

6 TREE-BASED RANDOMLY GENERATED FUNCTIONS

We further our research by shifting our focus towards function generators that can create test functions with similar landscape characteristics as to our automotive crash functions. Inspired by [Škvorc et al. 2021a], our experiments show that the function generator proposed by [Tian et al. 2020a] has great potential in creating test functions similar to the crash functions. Generally, the

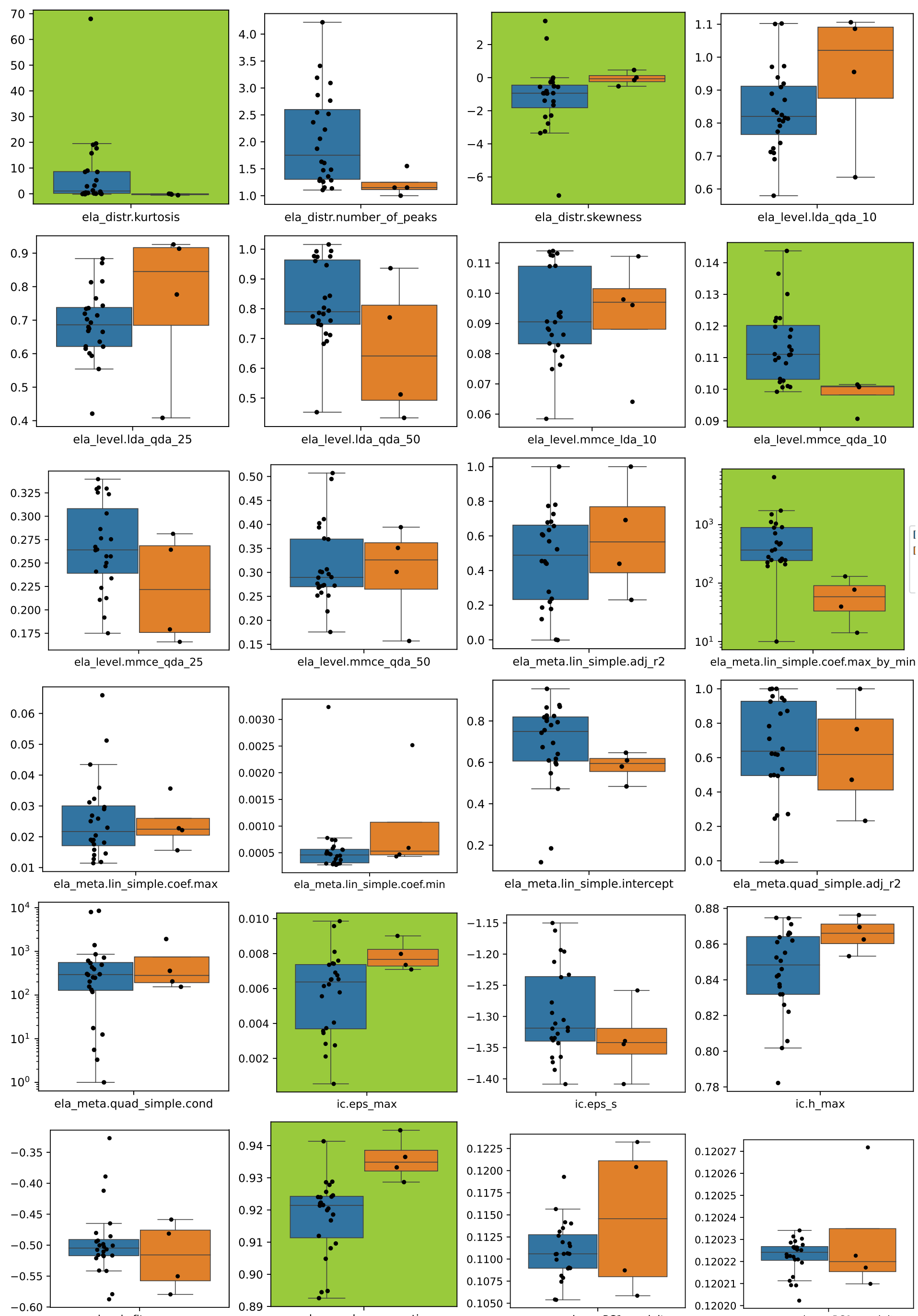


Fig. 5. Distribution of ELA features considered for clustering in raw feature values of the 24 BBOB functions (blue color) and four crash functions (orange color) of the problem instance Crash_2. An ELA feature is highlighted with green color, if the null hypothesis *distribution of ELA features between the crash and BBOB functions are similar* is rejected based on the KS-test with a confidence level of 95%.

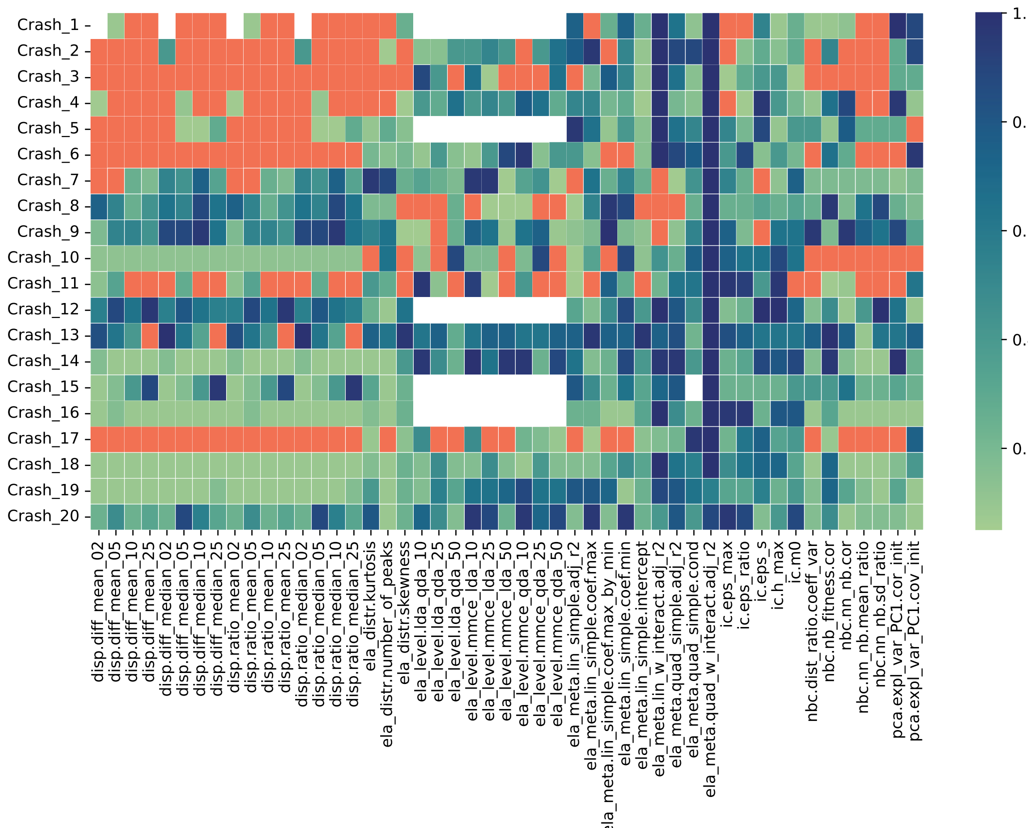


Fig. 6. Distribution comparison of all 49 ELA features between the crash and BBOB functions for all 20 automotive crash problems based on the p-value computed using the KS-test. A larger p-value (darker color) indicates that the null hypothesis *distribution of ELA features between the crash and BBOB functions are similar* is less likely to be rejected, while a smaller p-value (lighter color) for a higher chance of rejection. Red color indicates that the null hypothesis is rejected with a confidence level of 95%, that is, p-value less than 0.05, while white color indicates that ELA feature computation is skipped, for example, due to small sample size.

function generator works in a tree-based fashion, that is, by constructing and extending a function expression as a tree with mathematical operands and operators as leaf nodes, which are randomly selected from a predefined pool using a set of selection probability. We call these functions as tree-based randomly generated functions, or simply **randomly generated functions** (**RGF**), in the remainder of this article. Following this, we re-implement the function generator available at [Tian et al. 2020b] in Python, which was originally developed in Matlab, and integrate it into our pipeline in Figure 2 as test functions with some minor modifications, that is, a RGF is considered as invalid and discarded, if any of the following conditions is fulfilled:

(1) Error when converting the tree representation to an executable Python expression,
(2) Invalid objective values, for example, missing or infinity, and
(3) Small variance of objective values (< 1.0), to avoid on rare occasion a constant function due to rounding, for example, objective values are rounded off to a single integer.

Using the same experimental setup, we carry out another investigation on analyzing the landscape similarity between the crash functions and 1, 000 RGF. For instance, in the problem Crash_2,

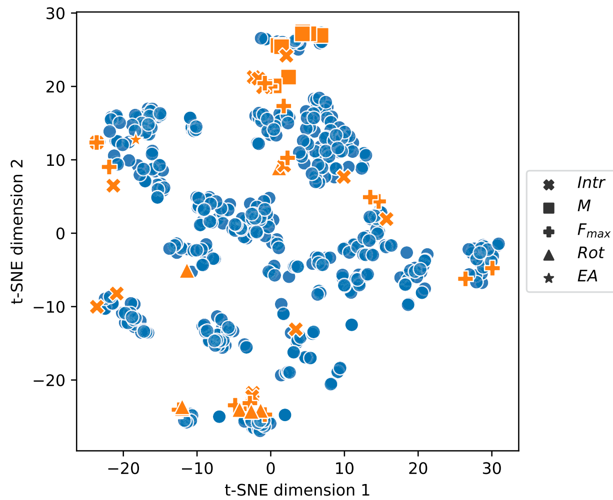


Fig. 7. Projection of the ELA feature space on a 2*d* visualization for all crash (altogether 48 functions; orange color) and BBOB functions (altogether $24 \times 20 = 480$ functions; blue color) using the t-SNE approach.

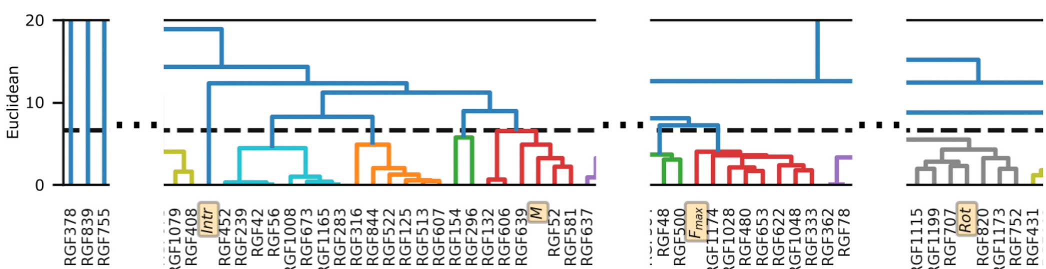


Fig. 8. Clustering pattern of the four crash functions and 1 000 RGF (labelled from RGF1 to RGF1000) for the problem instance Crash_2. Only relevant sections of the clustering pattern and y-axis are shown here due to the limited space. The labels of crash functions are highlighted in orange color, the reference Euclidean distance (same as in Figure 4) is marked with a dashed line, and clusters below the reference distance are assigned with different colors.

the mass, maximum force, and rotation function are clustered in the same group with several RGF at relatively smaller Euclidean distances as compared to the BBOB functions, as shown in Figure 8. For the intrusion function, however, no close neighbouring RGF can be identified. In cases like this, we suspect that either (i) such a similar function is not included in the set of test functions, which could be solved by expanding the function set, or (ii) creating such a similar RGF is simply not possible with the function generator in its current state, which is a limitation of our approach that makes further improvements necessary. The fact that increasing the number of RGF does not always lead to finding a more similar RGF indicates that the complexity of some problem classes is indeed insufficiently covered by the function generator. In general, as we can observe similar clustering patterns in most of the remaining automotive crash problems, we believe that we can find RGF that are representative for these crash problems in terms of landscape characteristics.

To visualize the distribution of functions across the ELA feature space, we project the high-dimensional ELA space on a 2*d* visualization using the t-SNE approach, as shown in Figure 9. In line with our interpretations in Section 5, many of the crash functions are separated away from

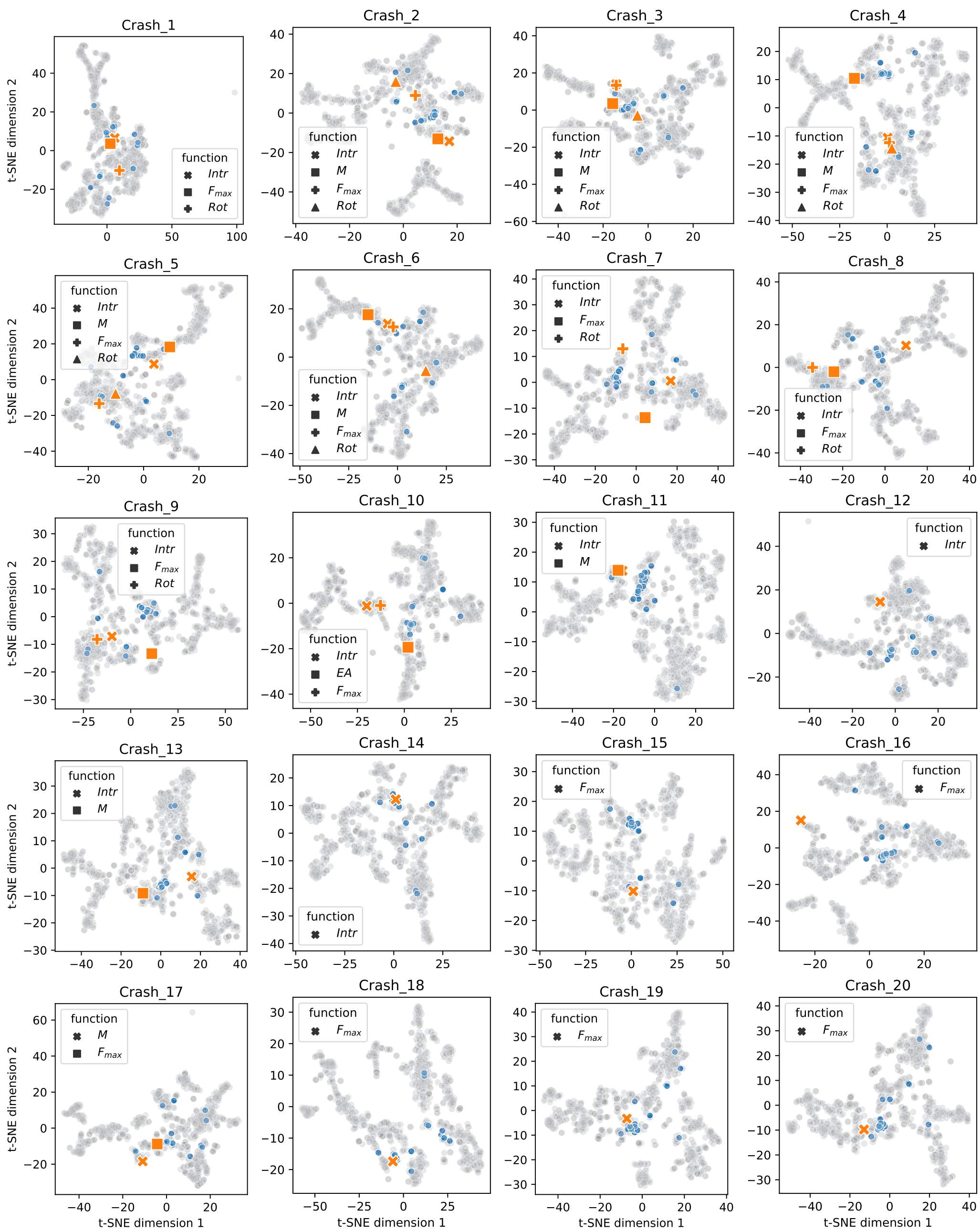


Fig. 9. Visualization of the ELA feature space on a $2d$ space for the crash functions (orange color), 24 BBOB functions (blue color), and 1, 000 RGF (gray color) are shown using the t-SNE approach for all 20 automotive crash problems.

the BBOB functions. On the other hand, they are closely clustered with some RGF. Following this, these neighbouring RGF have a higher landscape similarity as the crash functions, and thus can be considered as representative functions.

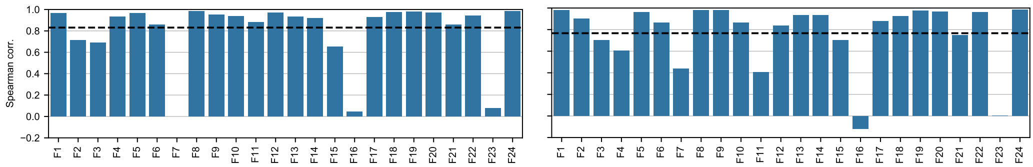


Fig. 10. Spearman's correlation based on the ranking of configuration performances between the BBOB and their representative functions. The mean correlation across all 24 BBOB functions is marked with a dashed line. Left: Optimization solution as metric; right: AUC as metric.

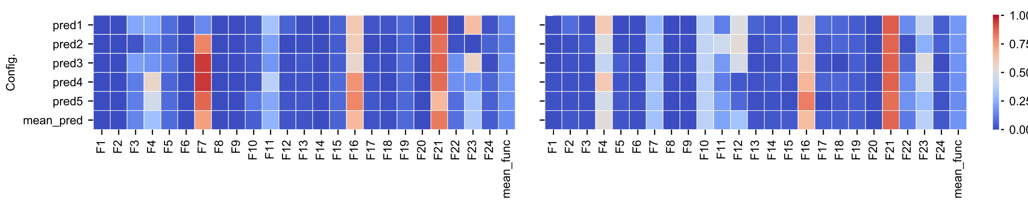


Fig. 11. Performance of the predicted best five configurations on the BBOB functions, re-scaled between the range of 0 (the best; blue color) and 1 (the worst; red color). The bottom row is added for the mean over all predicted configurations and the right column for the mean over all the BBOB functions. Left: Optimization solution as metric; right: AUC as metric.

7 REPRESENTATIVE FUNCTIONS FOR OPTIMIZATION PERFORMANCE PREDICTION

From a pool of 10, 000 RGF, we consider the RGF that has the most similar landscape characteristics, that is, the smallest difference in ELA features, to a BBOB function as the corresponding representative function, unless stated otherwise. Within the scope of this research, we consider only HPO without algorithm selection, by comparing the performance of algorithms with different configurations.

7.1 Optimization using CMA-ES

We rank each configuration based on its optimization performance, where the configuration with the best performance is assigned with rank one, and configurations having the same performance or ties are assigned with the same rank. For the comparison of configuration ranking, we consider the Spearman's correlation, which is informative regarding the overall tendency of configuration performances. A positive Spearman's correlation indicates that good configurations on representative functions can also perform well on the BBOB functions. As shown in Figure 10, we observe a strong positive correlation in configuration performances across most of the BBOB functions for both performance metrics, indicating that these representative functions can be exploited for optimization performance prediction. Nonetheless, in F7 (step ellipsoidal), F16 (Weierstrass), and F23 (Katsuura), the configuration performance correlations are rather weak, particularly the slightly negative correlation in the AUC metric in F16.

Eventually, we are mainly interested in identifying the best or several top configurations predicted on the representative functions and applying them on the actual BBO problem. Following this, we investigate further the competitiveness of the predicted best five configurations on the BBOB functions, as shown in Figure 11. In general, the results are similar to the Spearman's correlation, where the predicted best five configurations can perform well on most of the BBOB functions, except in the case of F7, F16, and F21 (Gallagher's Gaussian). While it is somewhat expected for F7

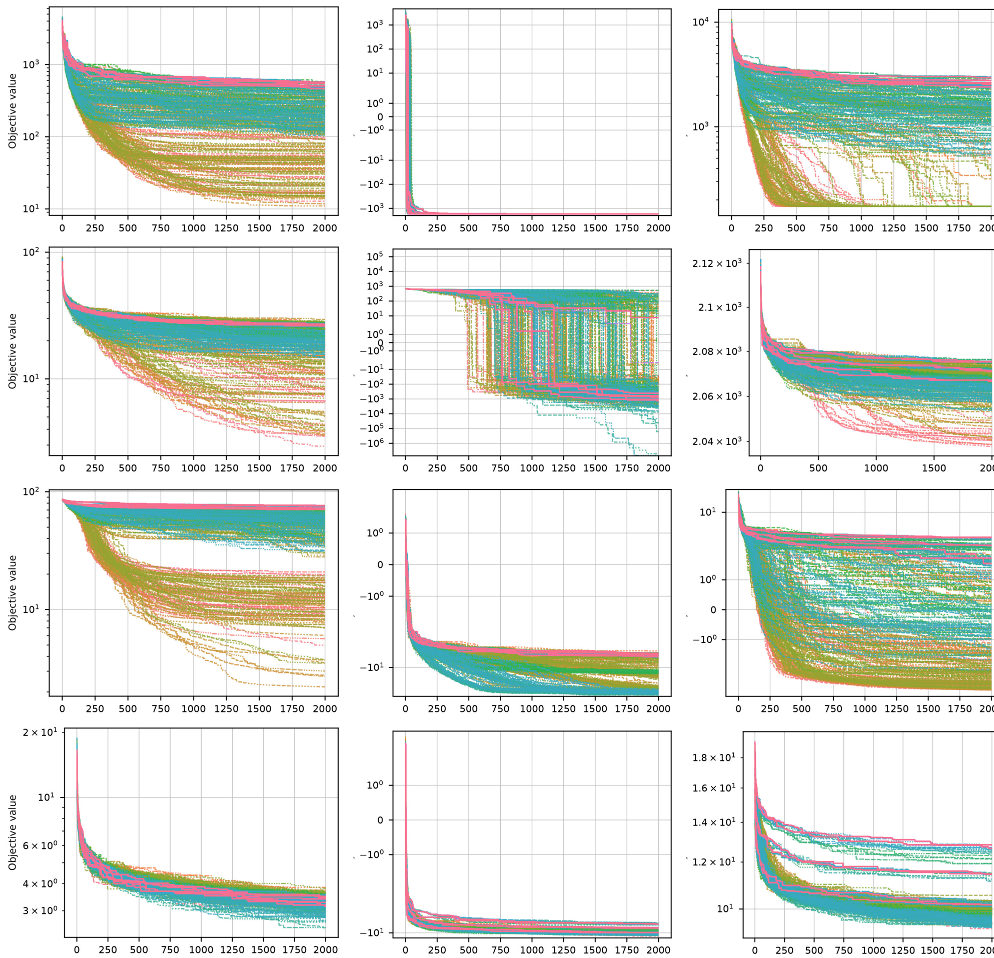
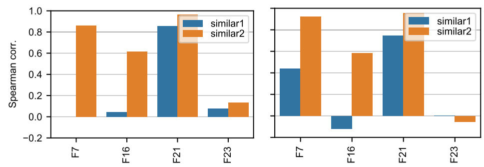


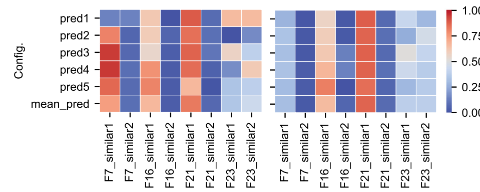
Fig. 12. The optimization runs of altogether 972 configurations for the BBOB function (left column), the most similar RGF (middle column) and the second most similar RGF (right column) for F7, F16, F21, and F23 (from top to bottom row). On the y-axis, the objective values in log-scale are shown, and the number of function evaluations on the x-axis. For the BBOB functions (left column), the objective values are re-scaled to 0 being the true global optimum. Each curve represents an optimization run (median of 30 repetitions) using a configuration and each color represents the same configuration.

and F16 due to the low ranking correlation, the relationship between the Spearman's correlation and performance of predicted best configurations is completely the opposite in F21 and F23.

To explain this observation, we examine the optimization runs of these four BBOB functions in-depth, as visualized in Figure 12. For instance, multiple configurations are assigned with the same rank in F7 and F23, since they all find the same optimization solution, resulting in an ambiguous ranking of configuration performances. The situation is more extreme in F7, where all the configurations are ranked as the "best". The situation is similar in F16, where the representative function has a global optimum that can only occasionally be found by the optimization runs, leading to an inconsistent configuration ranking. While the overall configuration performances are apparently comparable, the top configurations in F21 and its representative function seem to be different.



(a) Spearman's correlation of the ranking of configuration performances based on the optimization solution (left) and AUC (right) metric.



(b) Performance of the predicted best five configurations on the BBOB functions, re-scaled between the range of 0 (the best) and 1 (the worst), based on the optimization solution (left) and AUC (right) metric.

Fig. 13. Comparison of the representativeness between the most similar (labelled as "similar1") and second most similar RGF (labelled as "similar2") in F7, F16, F21, and F23.

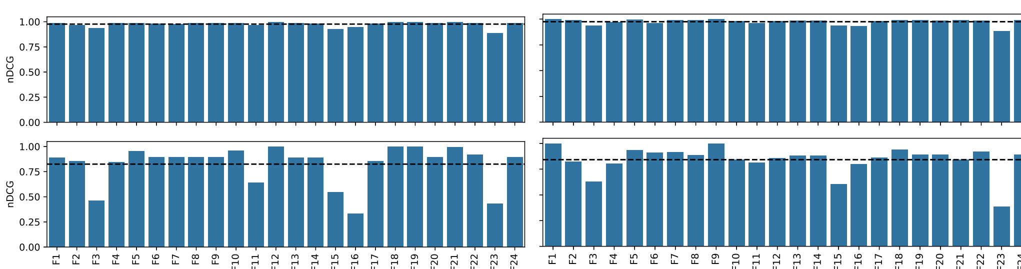
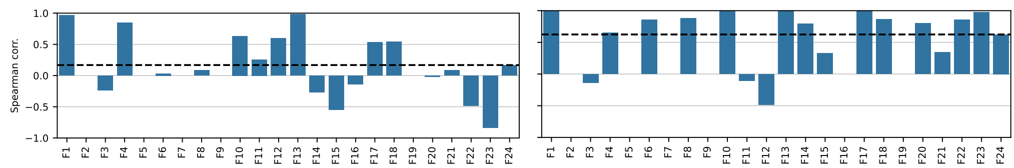


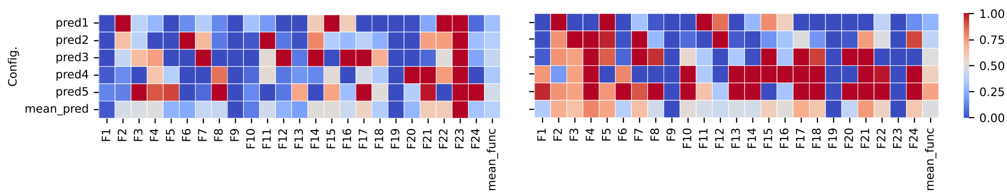
Fig. 14. Ranking quality of configurations based on their performances on the BBOB and representative functions using the nDCG method, with the second most similar RGF as representative functions in F7, F16, F21, and F23. The average score across all 24 BBOB functions is marked with a dashed line. Top row: Ranking of all configurations; bottom row: Ranking of only top five configurations; left column: Optimization solution as metric; right column: AUC as metric.

We further our investigation with the hypothesis that perhaps not all RGF are appropriate for HPO purposes, especially when a precise ranking of configuration performances is not possible. Subsequently, we repeat the experiments again for F7, F16, F21, and F23, but now using the RGF with the second most similar landscape characteristics as representative functions. As shown in Figures 12 and 13, we observe a promising improvement in most of the BBOB functions, both in the ranking correlation as well as the performance of predicted best configurations. Since using the second most similar RGF does not actually bring any improvement in F23, we suspect that the RGF are in fact insufficiently representative for this problem class, which needs to be improved in the future.

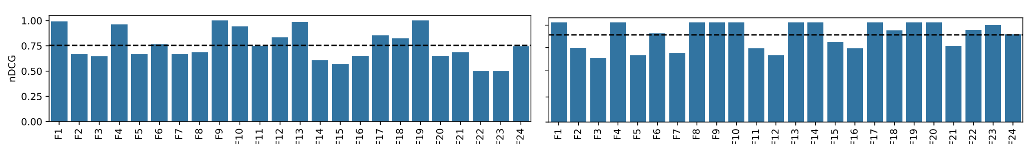
Additionally, we consider the **normalized discounted cumulative gain (nDCG)** [Järvelin and Kekäläinen 2002] to evaluate the ranking quality of configuration performances, which is commonly employed in the field of information retrieval. Essentially, the nDCG measures the quality of a ranking based on the respective relevance scores, where the best score 1.0 represents a perfect ranking. In this work, we analyze the ranking of configurations on the BBOB functions based on their ranking predicted on the representative functions as relevance scores. When all configurations are considered, the configuration ranking is good with a high nDCG value in all the BBOB functions, as visualized in Figure 14. On the contrary, when only the top five configurations are considered, the configuration ranking is below the average in some of the BBOB functions, particularly for the optimization solution metric in F3, F15, F16, and F23.



(a) Spearman's correlation based on the ranking of configuration performances between the BBOB and representative functions. The mean correlation across all 24 BBOB functions is marked with a dashed line.



(b) Performance of the predicted best five configurations on the BBOB functions, re-scaled between the range of 0 (the best) and 1 (the worst). The bottom row is added for the mean over all predicted configurations and the right column for the mean over all the BBOB functions.



(c) Ranking quality of the top five configurations based on their performances on the BBOB and representative functions using the nDCG method. The average score across all 24 BBOB functions is marked with a dashed line.

Fig. 15. Comparison of configuration performances using the BO based on the optimization solution (left column) and AUC (right column) metric between the BBOB and representative functions.

Following this, we conclude that our approach of using RGF with similar landscape characteristics as representative functions indeed has promising potential in predicting the general tendency of configuration performances on actual problems. However, there is still room for improvement in our approach, especially for the prediction of the best configuration or top configurations. We suspect that prediction of algorithm performances based on a single representative function might be not robust for all problems, and considering a set of representative functions can potentially improve the predictions.

7.2 Optimization using BO

The results using the BO are visualized in Figure 15 for the Spearman's correlation, performance of predicted best configurations and quality of configuration ranking based on the nDCG method. In general, the average Spearman's correlation in configuration ranking is much lower for BO than CMA-ES, especially in more complicated functions, for example, F24 (Lunacek bi-Rastrigin). On the other hand, the best predicted configuration on the representative function seems to be still competitive on most of the BBOB functions. In some of the BBOB functions, for example, F2 (Ellipsoidal), F5, F7, F9 (Rosenbrock), F12, F15 (Rastrigin), F16, and F23, the best possible optimization solution within the 300 evaluation budget for either the BBOB or representative function is included in the initial DoE utilized to build the GP models. Since all configurations can find the same solution, this eventually leads to an ambiguous ranking of configurations, as described in Section 7.1. One way of overcoming this, for instance, is by utilizing another sampling method, such as the **Latin hypercube sampling** (**LHS**) method, where the best possible solution is not included in the initial DoE. While the results are not shown in this work, we manage to improve the Spearman's correlation to around 0.9 in F5, when using the LHS method and optimization solution as metric in our testing. We suspect that the weak ranking correlation and poor performance of predicted best configurations could be also due to the fact that the number of configurations considered is low, which could be improved by expanding the hyperparameter set.

8 CONCLUSIONS AND FUTURE RESEARCH

In this article, we propose an approach for characterizing the problem class of real-world expensive BBO problems and identifying test functions that are appropriate as representative functions, which can be exploited to predict the optimization performance of an algorithm. In this context, we develop an automated pipeline based on 20 automotive crashworthiness optimization problem instances, which is aimed to be applicable on other expensive BBO domains with modifications. Precisely, we focus on analyzing crashworthiness optimization for different automotive crash scenarios based on FE simulations with between 8 and 22 thicknesses as design variables.

In our approach, we characterize the automotive crashworthiness problem instances by comparing their landscape properties or ELA features with those of some test functions, such as the well-established BBOB suite. By analyzing the ELA features, we can have a better understanding of the landscape characteristics of automotive crash problems. Through hierarchical clustering, we show that many of the crash functions are separated from the BBOB functions, indicating that the BBOB functions are insufficient to characterize these crash functions, which belong to a distinguishable problem class. Consequently, we further our investigation with a function generator, which is capable of randomly creating tree-based test functions with similar landscape properties as the crash functions. We suspect that a RGF with similar landscape characteristics can be identified for a particular BBO problem, provided that (i) the set of RGF is sufficiently large, and (ii) the problem complexity is well covered by the function generator.

Based on the BBOB suite and two optimization algorithms, namely the CMA-ES and BO algorithm, our results reveal that the RGF with similar landscape characteristics are indeed informative in predicting the algorithm performances on many of the BBOB functions. In other words, the algorithm performances are comparable between the BBOB and similar RGF, where top configurations on the RGF can perform well on the BBOB functions. Therefore, we propose to consider such RGF as scalable and fast-to-evaluate representatives of real-world problem instances, and use them to predict the expected algorithm performances on actual BBO problems. Since the BBOB suite covers a wide range of problem classes, we believe that our approach can generalize well to other BBO domains of similar problem classes.

In the future, we plan to continue developing the proposed pipeline for designing efficient optimization algorithms for real-world expensive BBO problems based on some representative functions, refer to Figure 1. Precisely, we would like to (i) validate the findings of this work on real-world automotive crash problems, using FE simulations for function evaluation, and (ii) investigate the potential of RGF as representative functions for automated algorithm selection and HPO, for example, using the **sequential model-based algorithm configuration** (**SMAC**) [Lindauer et al. 2022]. With this pipeline, we aim to improve the overall optimization efficiency by specifically designing and fine-tuning optimization algorithms for real-world BBO problems. Ideally, we would like to extend our approach for multi-objective and constrained optimization problems.

Another interesting research direction could be investigating the representativeness of data-driven surrogate models trained on the DoE samples in predicting algorithm performances and comparing them to our approach, as briefly mentioned in Section 2. In this context, we would like to better understand the impact of DoE sample size as a function of problem dimensionality on the representativeness of functions identified as representative functions. Furthermore, we would like to raise our concerns regarding the effectiveness of ELA features in quantifying the similarities between function landscapes, since the ELA features are manually engineered by experts and thus might be biased in feature computation. Accordingly, another research outlook could be exploring other alternatives for feature computation in an unbiased way [van Stein et al. 2023]. In the meanwhile, we plan to explore (i) other methods in selecting optimal set of ELA features, (ii) distance metrics, for example, the Malahanobis distance, to reliably quantify the landscape similarities between functions, and (iii) methods in normalizing the ELA features before distance computation [Rakotoarison et al. 2022].

Currently, the complexity of RGF is mainly limited by the predefined pool of mathematical operands and operators. An interesting idea for future work could be extending the diversity of RGF that can be generated, for example, jump functions, to characterize a broader spectrum of optimization problems. Rather than using the random search approach, as the function generation in its current state, we would like to investigate the potential of creating test functions with specific landscape characteristics in our pipeline, for example, evolving test functions using genetic programming [Long et al. 2023a], which could potentially overcome the problem, when no similar RGF can be created by the function generator.

REPRODUCIBILITY STATEMENT

Due to the limited space, our experimental data, for example, ELA feature values and subset of selected ELA features, remaining results and figures not included in this article are made available in our repository in https://doi.org/10.5281/zenodo.10397153.

REFERENCES

Altair Engineering Inc. 2022. Altair HyperStudy. Retrieved January 15, 2022 from https://www.altair.com/hyperstudy/

Carolin Benjamins, Anja Jankovic, Elena Raponi, Koen van der Blom, Marius Lindauer, and Carola Doerr. 2022. Towards Automated Design of Bayesian Optimization via Exploratory Landscape Analysis. arXiv:2211.09678. Retrieved from https://arxiv.org/abs/2211.09678

Bernd Bischl, Olaf Mersmann, Heike Trautmann, and Mike Preuß. 2012. Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation* . Association for Computing Machinery, New York, NY, USA, 313–320. DOI: https://doi.org/10.1145/2330163.2330209

Jakob Bossek, Carola Doerr, Pascal Kerschke, Aneta Neumann, and Frank Neumann. 2020. Evolving sampling strategies for one-shot optimization tasks. In *Proceedings of the Parallel Problem Solving from Nature – PPSN XVI*, Thomas Bäck, Mike Preuss, André Deutz, Hao Wang, Carola Doerr, Michael Emmerich, and Heike Trautmann (Eds.), Vol. 12269. Springer International Publishing, Cham, 111–124. DOI: https://doi.org/10.1007/978-3-030-58112-1_8

Olivier Bousquet, Sylvain Gelly, Karol Kurach, Olivier Teytaud, and Damien Vincent. 2017. Critical Hyper-Parameters: No Random, No Cry. arXiv:1706.03200. Retrieved from https://arxiv.org/abs/1706.03200

Catharina Czech, Arne Kaps, and Fabian Duddeck. 2022. Robust multi-fidelity optimization approach exploiting data-driven, non-linear model order reduction. In *Proceedings of the 8th International Symposium on Reliability Engineering and Risk Management.* 357–363.

Jacob de Nobel, Diederick Vermetten, Hao Wang, Carola Doerr, and Thomas Bäck. 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.* Association for Computing Machinery, New York, NY, USA, 1375–1384. DOI: https://doi.org/10.1145/3449726.3463167

Konstantin Dietrich and Olaf Mersmann. 2022. Increasing the diversity of benchmark function sets through affine recombination. In *Proceedings of the Parallel Problem Solving from Nature – PPSN XVII: 17th International Conference.* Springer-Verlag, Berlin, 590–602. DOI: https://doi.org/10.1007/978-3-031-14714-2_41

Carola Doerr, Johann Dreo, and Pascal Kerschke. 2019. Making a case for (hyper-)parameter tuning as benchmark problems. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion.* Association for Computing Machinery, New York, NY, USA, 1755–1764. DOI: https://doi.org/10.1145/3319619.3326857

Carola Doerr, Hao Wang, Furong Ye, Sander van Rijn, and Thomas Bäck. 2018. IOHprofiler: A benchmarking and profiling tool for iterative optimization heuristics. arXiv:1810.05281. Retrieved from https://arxiv.org/abs/1810.05281

Johann Dréo, Carola Doerr, and Yann Semet. 2019. Coupling the design of benchmark with algorithm in landscape-aware solver design. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. Association for Computing Machinery, New York, NY, USA, 1419–1420. DOI: https://doi.org/10.1145/3319619.3326821

Fabian Duddeck. 2008. Multidisciplinary optimization of car bodies. *Structural and Multidisciplinary Optimization* 35, 4 (2008), 375–389. DOI: https://doi.org/10.1007/s00158-007-0130-6

H. Fang, Masoud Rais-Rohani, Z. Liu, and Mark Horstemeyer. 2005. A comparative study of metamodeling methods for multiobjective crashworthiness optimization. *Computers & Structures* 83, 25-26 (2005), 2121–2136. DOI: https://doi.org/10.1016/j.compstruc.2005.02.025

Jianguang Fang, Guangyong Sun, Na Qiu, Nam-Ho Kim, and Qing Li. 2017. On design optimization for structural crashworthiness and its state of the art. *Structural and Multidisciplinary Optimization* 55, 3 (2017), 1091–1119. DOI: https://doi.org/10.1007/s00158-016-1579-y

Jimmy Forsberg and Larsgunnar Nilsson. 2006. Evaluation of response surface methodologies used in crashworthiness optimization. *International Journal of Impact Engineering* 32, 5 (2006), 759–777. DOI: https://doi.org/10.1016/j.ijimpeng.2005.01.007

Karim Hamza and Mohamed Shalaby. 2014. A framework for parallelized efficient global optimization with application to vehicle crashworthiness optimization. *Engineering Optimization* 46, 9 (2014), 1200–1221. DOI: https://doi.org/10.1080/0305215X.2013.827672

Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. 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, 1689–1696. DOI: https://doi.org/10.1145/1830761.1830790

Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tušar, and Dimo Brockhoff. 2021. COCO: A platform for comparing continuous optimizers in a black-box setting. *Optimization Methods and Software* 36, 1 (2021), 114–144. DOI: https://doi.org/10.1080/10556788.2020.1808977

Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. 2009. *Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions*. Research Report RR-6829. INRIA. Retrieved from https://hal.inria.fr/inria-00362633

Nikolaus Hansen and Andreas Ostermeier. 1996. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the IEEE International Conference on Evolutionary Computation*. IEEE, 312–317.

Anja Jankovic and Carola Doerr. 2020. Landscape-aware fixed-budget performance regression and algorithm selection for modular CMA-ES variants. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference* . Association for Computing Machinery, New York, NY, USA, 841–849. DOI: https://doi.org/10.1145/3377930.3390183

Anja Jankovic, Gorjan Popovski, Tome Eftimov, and Carola Doerr. 2021. *The Impact of Hyper-Parameter Tuning for Landscape-Aware Performance Regression and Algorithm Selection*. In *Proceedings of the Genetic and Evolutionary Computation Conference*. Association for Computing Machinery, New York, NY, USA, 687–696. DOI: https://doi.org/10.1145/3449639.3459406

Tomas Jansson, Larsgunnar Nilsson, and Marcus Redhe. 2003. Using surrogate models and response surfaces in structural optimization – with application to crashworthiness design and sheet metal forming. *Structural and Multidisciplinary Optimization* 25, 2 (2003), 129–140. DOI: https://doi.org/10.1007/s00158-002-0279-y

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20, 4 (october 2002), 422–446. DOI: https://doi.org/10.1145/582415.582418

Donald R. Jones, Matthias Schonlau, and William J. Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization* 13, 4 (1998), 455.

Arne Kaps, Catharina Czech, and Fabian Duddeck. 2022. A hierarchical kriging approach for multi-fidelity optimization of automotive crashworthiness problems. *Structural and Multidisciplinary Optimization* 65, 4 (april 2022), 15 pages. DOI: https://doi.org/10.1007/s00158-022-03211-2

Pascal Kerschke, Holger Hoos, Frank Neumann, and Heike Trautmann. 2019. Automated algorithm selection: Survey and perspectives. *Evolutionary Computation* 27, 1 (2019), 3–45. DOI: https://doi.org/10.1162/evco_a_00242

Pascal Kerschke, Mike Preuss, Simon Wessing, and Heike Trautmann. 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, 265–272. DOI: https://doi.org/10.1145/2739480.2754642

Pascal Kerschke and Heike Trautmann. 2019a. Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning. *Evolutionary Computation* 27, 1 (2019), 99–127. DOI: https://doi.org/10.1162/evco_a_00236

Pascal Kerschke and Heike Trautmann. 2019b. *Comprehensive Feature-Based Landscape Analysis of Continuous and Constrained Optimization Problems Using the R-Package Flacco*. Springer International Publishing, Cham, 93–123. DOI: https://doi.org/10.1007/978-3-030-25147-5_7

Pascal Kerschke and Heike Trautmann. 2019c. flacco: Feature-Based Landscape Analysis of Continuous and Constrained Optimization Problems. Retrieved January 15, 2022 from https://github.com/kerschke/flacco

Morteza Kiani, Imtiaz Gandikota, A. Parrish, Keiichi Motoyama, and Masoud Rais-Rohani. 2013. Surrogate-based optimisation of automotive structures under multiple crash and vibration design criteria. *International Journal of Crashworthiness* 18, 5 (2013), 473–482. DOI: https://doi.org/10.1080/13588265.2013.805294

Hasan Kurtaran, Azim Eskandarian, D. Marzougui, and N. Bedewi. 2002. Crashworthiness design optimization using successive response surface approximations. *Computational Mechanics* 29, 4 (2002), 409–421. DOI: https://doi.org/10.1007/s00466-002-0351-x

Xingtao Liao, Qing Li, Xujing Yang, Wei Li, and Weigang Zhang. 2008. A two-stage multi-objective optimisation of vehicle crashworthiness under frontal impact. *International Journal of Crashworthiness* 13, 3 (2008), 279–288. DOI: https://doi.org/10.1080/13588260801933659

Marius Lindauer, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter. 2022. SMAC3: A versatile bayesian optimization package for hyperparameter optimization. *Journal of Machine Learning Research* 23, 54 (2022), 1–9. Retrieved from http://jmlr.org/papers/v23/21-0888.html

Livermore Software Technology Corporation. 2019. LS-DYNA Theory Manual. Retrieved December 28, 2021 from https://ftp.lstc.com/anonymous/outgoing/jday/manuals/DRAFT_Theory.pdf

Fu Xing Long, Bas van Stein, Moritz Frenzel, Peter Krause, Markus Gitterle, and Thomas Bäck. 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, 1227–1236. DOI: https://doi.org/10.1145/3512290.3528712

Fu Xing Long, Diederick Vermetten, Anna V. Kononova, Roman Kalkreuth, Kaifeng Yang, Thomas Bäck, and Niki van Stein. 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, 119–130. DOI: https://doi.org/10.5220/0012206200003595

Fu Xing Long, Diederick Vermetten, Bas van Stein, and Anna V. Kononova. 2023b. BBOB instance analysis: Landscape properties and algorithm performance across problem instances. In *Proceedings of the Applications of Evolutionary Computation: 26th European Conference, EvoApplications 2023, Held as Part of EvoStar 2023*. Springer-Verlag, Berlin, 380–395. DOI: https://doi.org/10.1007/978-3-031-30229-9_25

Monte Lunacek and Darrell Whitley. 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, 477–484. DOI: https://doi.org/10.1145/1143997.1144085

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

Frank J. Massey Jr. 1951. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association* 46, 253 (1951), 68–78.

Olaf Mersmann, Bernd Bischl, Heike Trautmann, Mike Preuss, Claus Weihs, and Günter Rudolph. 2011. Exploratory landscape analysis. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation* . Association for Computing Machinery, New York, NY, USA, 829–836. DOI: https://doi.org/10.1145/2001576.2001690

Olaf Mersmann, Mike Preuss, and Heike Trautmann. 2010. Benchmarking evolutionary algorithms: Towards exploratory landscape analysis. In *Proceedings of the Parallel Problem Solving from Nature, PPSN XI*, Robert Schaefer, Carlos Cotta, Joanna Kołodziej, and Günter Rudolph (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 73–82. DOI: https://doi.org/10.1007/978-3-642-15844-5_8

Jonas Mockus. 1982. The Bayesian approach to global optimization. In *Proceedings of the System Modeling and Optimization*. Springer, 473–481.

Mario Andrés Muñoz, Michael Kirley, and Kate Smith-Miles. 2022. Analyzing randomness effects on the reliability of exploratory landscape analysis. *Natural Computing: An International Journal* 21, 2 (june 2022), 131–154. DOI: https://doi.org/10.1007/s11047-021-09847-1

Mario Andrés Muñoz, Michael Kirley, and Saman K. Halgamuge. 2015a. Exploratory landscape analysis of continuous space optimization problems using information content. *IEEE Transactions on Evolutionary Computation* 19, 1 (2015), 74–87. DOI: https://doi.org/10.1109/TEVC.2014.2302006

Mario Andrés Muñoz, Yuan Sun, Michael Kirley, and Saman K. Halgamuge. 2015b. Algorithm selection for black-box continuous optimization problems: A survey on methods and challenges. *Information Sciences* 317 (2015), 224–245. DOI: https://doi.org/10.1016/j.ins.2015.05.010

Fionn Murtagh and Pedro Contreras. 2012. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 2, 1 (2012), 86–97. DOI: https://doi.org/10.1002/widm.53

Fernando Nogueira. 2014. Bayesian Optimization: Open source constrained global optimization tool for Python. Retrieved from https://github.com/fmfn/BayesianOptimization

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research* 12, 85 (2011), 2825–2830. Retrieved from http://jmlr.org/papers/v12/pedregosa11a.html

Maxim Pikalov and Vladimir Mironovich. 2021. *Automated Parameter Choice with Exploratory Landscape Analysis and Machine Learning*. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. Association for Computing Machinery, New York, NY, USA, 1982–1985. DOI: https://doi.org/10.1145/3449726.3463213

Raphael Patrick Prager. 2022. pflacco: The R-package flacco in native Python code. Retrieved May 03, 2023 from https://github.com/Reiyan/pflacco

Raphael Patrick Prager and Heike Trautmann. 2023a. Nullifying the inherent bias of non-invariant exploratory landscape analysis features. In *Proceedings of the Applications of Evolutionary Computation: 26th European Conference, EvoApplications 2023, Held as Part of EvoStar 2023*. Springer-Verlag, Berlin, 411–425. DOI: https://doi.org/10.1007/978-3-031-30229-9_27

Raphael Patrick Prager and Heike Trautmann. 2023b. Pflacco: Feature-based landscape analysis of continuous and constrained optimization problems in python. *Evolutionary Computation* (July 2023), 1–25. DOI: https://doi.org/10.1162/evco_a_00341

Herilalaina Rakotoarison, Louisot Milijaona, Andry Rasoanaivo, Michèle Sebag, and Marc Schoenauer. 2022. Learning metafeatures for automl. In *Proceedings of the International Conference on Learning Representations*.

Quentin Renau, Carola Doerr, Johann Dreo, and Benjamin Doerr. 2020. Exploratory landscape analysis is strongly sensitive to the sampling strategy. In *Proceedings of the Parallel Problem Solving from Nature – PPSN XVI*, Thomas Bäck, Mike Preuss, André Deutz, Hao Wang, Carola Doerr, Michael Emmerich, and Heike Trautmann (Eds.). Springer International Publishing, Cham, 139–153. DOI: https://doi.org/10.1007/978-3-030-58115-2_10

Quentin Renau, Johann Dreo, Carola Doerr, and Benjamin Doerr. 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, 2048–2051. DOI: https://doi.org/10.1145/3319619.3326913

Quentin Renau, Johann Dreo, Carola Doerr, and Benjamin Doerr. 2021. Towards explainable exploratory landscape analysis: Extreme feature selection for classifying BBOB functions. In *Proceedings of the Applications of Evolutionary Computation*, Pedro A. Castillo and Juan Luis Jiménez Laredo (Eds.). Springer International Publishing, Cham, 17–33. DOI: https://doi.org/10.1007/978-3-030-72699-7_2

John R. Rice. 1976. The algorithm selection problem. Advances in Computers, Vol. 15. Elsevier, 65–118. DOI: https://doi.org/10.1016/S0065-2458(08)60520-3

Lei Shi, Ren-Jye Yang, and Ping Zhu. 2012. A method for selecting surrogate models in crashworthiness optimization. *Structural and Multidisciplinary Optimization* 46, 2 (2012), 159–170. DOI: https://doi.org/10.1007/s00158-012-0760-1

David Simoncini, Sophie Barbe, Thomas Schiex, and Sébastien Verel. 2018. Fitness landscape analysis around the optimum in computational protein design. In *Proceedings of the Genetic and Evolutionary Computation Conference* . Association for Computing Machinery, New York, NY, USA, 355–362. DOI: https://doi.org/10.1145/3205455.3205626

Kate Smith-Miles and Mario Andrés Muñoz. 2023. Instance space analysis for algorithm testing: Methodology and software tools. *ACM Computing Surveys* 55, 12 (march 2023), 31 pages. DOI: https://doi.org/10.1145/3572895

Il'ya Meerovich Sobol'. 1967. On the distribution of points in a cube and the approximate evaluation of integrals. *U. S. S. R. Computational Mathematics and Mathematical Physics* 7, 4 (1967), 86–112. DOI: https://doi.org/10.1016/0041-5553(67)90144-9

Guangyong Sun, Ye Tian, Ruoyu Wang, Jianguang Fang, and Qing Li. 2020. Parallelized multiobjective efficient global optimization algorithm and its applications. *Structural and Multidisciplinary Optimization* 61, 2 (2020), 763–786. DOI: https://doi.org/10.1007/s00158-019-02417-1

André Thomaser, Anna V. Kononova, Marc-Eric Vogt, and Thomas Bäck. 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* (Boston, Massachusetts). Association for Computing Machinery, New York, NY, USA, 2036–2045. DOI: https://doi.org/10.1145/3520304.3533979

Ye Tian, Shichen Peng, Xingyi Zhang, Tobias Rodemann, Kay Chen Tan, and Yaochu Jin. 2020b. Algorithm-Recommendation. Retrieved January 15, 2022 from https://github.com/BIMK/Algorithm-Recommendation

Ye Tian, Shichen Peng, Xingyi Zhang, Tobias Rodemann, Kay Chen Tan, and Yaochu Jin. 2020a. A recommender system for metaheuristic algorithms for continuous optimization based on deep recurrent neural networks. *IEEE Transactions on Artificial Intelligence* 1, 1 (2020), 5–18. DOI: https://doi.org/10.1109/TAI.2020.3022339

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. Retrieved from http://jmlr.org/papers/v9/vandermaaten08a.html

Bas van Stein, Fu Xing Long, Moritz Frenzel, Peter Krause, Markus Gitterle, and Thomas Bäck. 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, 515–518. DOI: https://doi.org/10.1145/3583133.3590609

Bas van Stein, Hao Wang, and Thomas Bäck. 2020. Neural network design: Learning from neural architecture search. In *Proceedings of the 2020 IEEE Symposium Series on Computational Intelligence*. IEEE, 1341–1349. DOI: https://doi.org/10.1109/SSCI47803.2020.9308394

Diederick Vermetten, Furong Ye, Thomas Bäck, and Carola Doerr. 2023b. MA-BBOB: Many-affine combinations of BBOB functions for evaluating AutoML approaches in noiseless numerical black-box optimization contexts. (2023). Retrieved from https://openreview.net/forum?id=uN70Dum6pC2

Diederick Vermetten, Furong Ye, and Carola Doerr. 2023a. Using affine combinations of BBOB problems for performance assessment. In *Proceedings of the Genetic and Evolutionary Computation Conference*. Association for Computing Machinery, New York, NY, USA, 8730–881. DOI: https://doi.org/10.1145/3583131.3590412

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods* 17 (2020), 261–272. DOI: https://doi.org/10.1038/s41592-019-0686-2

Urban Škvorc, Tome Eftimov, and Peter Korošec. 2020. Understanding the problem space in single-objective numerical optimization using exploratory landscape analysis. *Applied Soft Computing* 90 (2020), 106138. DOI: https://doi.org/10.1016/j.asoc.2020.106138

Urban Škvorc, Tome Eftimov, and Peter Korošec. 2021a. *A Complementarity Analysis of the COCO Benchmark Problems and Artificially Generated Problems*. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. Association for Computing Machinery, New York, NY, USA, 215–216. DOI: https://doi.org/10.1145/3449726.3459585

Urban Škvorc, Tome Eftimov, and Peter Korošec. 2021b. The effect of sampling methods on the invariance to function transformations when using exploratory landscape analysis. In *Proceedings of the 2021 IEEE Congress on Evolutionary Computation*. 1139–1146. DOI: https://doi.org/10.1109/CEC45853.2021.9504739

Joe H. Ward Jr. 1963. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association* 58, 301 (1963), 236–244. DOI: https://doi.org/10.1080/01621459.1963.10500845

Received 30 November 2022; revised 17 December 2023; accepted 24 January 2024