

DATA-DRIVEN PROBLEM CLASSIFICATION AND ALGORITHM SELECTION FOR INJECTION MOLDING OPTIMIZATION

Andreas Fischbach^{1*}, Frederik Rehbach², Denis Anders¹

¹ Cologne University of Applied Sciences (TH Köln), Group for Computational Mechanics and Fluid Dynamics, Steinmüllerallee 1, 51643 Gummersbach, Germany

² Cologne University of Applied Sciences (TH Köln), Institute for Data Science, Engineering, and Analytics, Steinmüllerallee 1, 51643 Gummersbach, Germany

Abstract

Injection Molding (IM) is an established and essential production process. Dynamic process changes according to market demands, and advances in additive manufacturing technologies enable both chances and challenges. New molds can be produced more quickly and cheaper, but the demand for ongoing parameter optimization due to the changes in the production process rises. Digital-Twin-based process simulation assists process engineers in estimating optimal parameter settings. Due to its complexity, manual optimization is inefficient. Selecting an appropriate optimization algorithm for this task is challenging. There is a need for automated data science pipelines, especially for small and medium-sized enterprises (SMEs) that often lack algorithm engineering and data science capabilities. This paper addresses the data-driven classification of different IM optimization problems retrieved by IM simulation. As these IM simulations are computationally expensive, we apply Gaussian Process Simulations (GPS) on a small number of samples of the IM simulations to compute cheap to evaluate test problems. We compare a single and a 2-stage variant of GPS, both in their conditional and unconditional forms. The single variant tends to preserve the global characteristics of the problem. The 2-stage variant adds a second simulation model based on the residuals of the first model to simulate the finer local structure of the problem. The retrieved test problems are classified using Exploratory Landscape Analysis (ELA) applied to several minimization objectives of the IM process: max. volume shrinkage [%], avg. volume shrinkage [%], required cooling time [s], max. warpage [mm], max. deformation [mm], and avg. shrinkage [mm]. Finally, benchmark experiments will be conducted to compare optimization algorithms on the test problems and relate the results to the retrieved ELA features. These results build the foundation for algorithm selection and online parameter optimization.

Keywords: Injection molding, gaussian process regression, simulation, parameter optimization, exploratory landscape analysis

1. INTRODUCTION

Injection molding (IM) is among the most essential industrial polymer processing techniques. It is widely used in commercial and consumer products due to its high level of automation and reliability and its ability to produce complex shapes in relatively short production cycles. The IM process typically consists of the following steps. 1) Filling granulate in a barrel via a hopper, 2) melting the filled polymer using heaters, 3) mixing the polymer using a screw, 4) injecting the molten polymer into a mold cavity, 5) cooling the molten polymer inside the mold cavity, and 6) ejecting the solidified part(s). These steps are driven by a set of parameter settings that should be optimized with respect to several criteria, e.g., the quality of the produced parts (in terms of dimensional accuracy, surface gloss, mechanical strength, etc.), energy consumption, and the cycle time, which also determines the production cost.

The location (i.e., parameter values) of the optima of these objectives may change according to the used material and its properties (e.g., moisture) or the cavities in the mold, which can nowadays be altered more easily via new technologies like additive manufacturing.

Classical optimization procedures like the design of experiments typically involve some down-time, e.g., when the quality of produced parts is measured, and further experiments must be planned. This paper focuses on analyzing online optimization methods employing simulations of the injection molding

process. These simulations allow an estimate of several objectives at a time in a single evaluation but also come with some drawbacks, as they typically are computationally expensive. The evaluation of a simulation may take much longer than an injection molding production cycle.

We will cope with this issue by employing surrogate models of these expensive simulations using Gaussian Process Models (GPM, Forrester, Söbester & Keane 2008) for interpolation and regression. Additionally, we will use implementations that allow for variations of these models, preserving characteristics of the underlying process (i.e., the injection molding simulation). These variations, further denoted as IM test instances, are computationally cheap and easy to evaluate. Additionally, they can be treated like functions, i.e., they can be evaluated at arbitrary search points (i.e., parameter settings). Therefore, they allow for benchmarking of different optimization algorithms and online algorithm selection and configuration. We especially consider two variants of Gaussian Process Simulations (GPS), a single-stage (1stage) and a two-stage (2stage) variant. The former tends to preserve the global structure of the process under investigation but might also lead to somewhat immoderately smoothed objectives. The latter adds a second GPM, which is fitted to the residuals of the first model, to fit the finer local structures of the objectives.

To ensure the validity of the test instances, we will analyze their objective landscapes using Exploratory landscape analysis (ELA, Mersmann et al. 2011). ELA covers several numerical features of optimization problem instances, e.g., describing their structure and distribution of objective function values, and allows for their classification. We employ ELA to visually classify and compare a) injection molding simulation instances, b) GPM-based (simulation) test instances, and c) artificial test instances, which are typically used for benchmarking continuous optimization algorithms. This leads to the main research questions addressed by our work:

- (RQ-1) Are GPMs leading to relevant test instances for the IM simulation?
- (RQ-2) What is the necessary (minimum) number of evaluations of the IM simulation to train the test instance generator for meaningful instances?
- (RQ-3) How do the IM test instances under investigation behave w.r.t. their type and parametrization, i.e., 1stage or 2stage, the number of cosines configured for spectral simulation, and the (un-) conditioning to the training data?

2. MATERIAL AND METHODS

2.1. Injection Molding Simulation

We employ Cadmould 3D-F to simulate the injection molding process in this contribution. In Cadmould, a model is set up to analyze the filling and cooling process and retrieve objective values regarding the quality, i.e., shrinkage, warpage, and deformation. Additionally, the required cooling time is measured, determining the cycle time. We focus on the melt temperature, holding pressure time, and cooling time in the mold for the control parameters. The control parameters and their ranges are listed in Table 1. For a list of the objectives, we refer to Table 2.

Table 1. Control parameter of the injection molding process

Name	Range	Description / Unit
x1	[230, 250]	melt temperature / °C
x2	[0.3, 6.3]	holding pressure time / s
x3	[10, 50]	cooling time in the mold after the filling / s

Table 2. Objectives evaluated by the injection molding simulation

Name	Objective	Description / Unit
y1	2.02	element-related mass / g
y2	minimize	max. volume shrinkage / %
y3	minimize	avg volume shrinkage / %
y4	minimize	required cooling time / s
y5	minimize	max. warpage / mm
y6	minimize	max. deformation / mm
y7	minimize	avg. shrinkage / mm

2.2. Gaussian Process Simulation

To fill the gap of missing real-world relevant optimization test problems, Fischbach et al. (2016) established a data-driven approach for test instance generation, consisting of the following steps:

1. Collect data consisting of a design matrix X and a vector Y of corresponding evaluations of a process.
2. Use (X,Y) to train a Gaussian process model (GPM), denoted as level 0 model.
3. Define a parameter vector α to perturb the level 0 model within defined bounds.
4. Apply α to the level 0 model to retrieve the desired number of test instances, further denoted as level 1 models, that preserve (some) characteristics of the underlying real-world process.

For details regarding GPM, we refer to Forrester, Sóbester & Keane (2008). In this work, we apply the process (see Fig. 1) of data-driven test instance generation using Gaussian process simulation (GPS), which not only varies the model parameters for simulations but preserves the covariance features of the fitted GPM. GPS allows for conditional and unconditional simulations. Both preserve the (covariance) structure of the model. Conditional simulations are additionally bound to the real observations Y . Zaefferer et al. (2017) applied GPS in the discrete domain for Benchmarking using matrix decomposition. This approach was extended to the continuous domain using spectral simulation by Zaefferer & Rehbach (2020). We use the implementation of the COBBS package for the R platform by Zaefferer (2020), which introduces a parameter $N_{\cos}$ for the number of cosines to be used for spectral simulation. Consequently, we define the perturbation vector $\alpha = (N_{\cos}, \text{conditional})$, where $N_{\cos}$ denotes the number of cosines, and conditional is a logical value, indicating the use of the conditional or unconditional case.

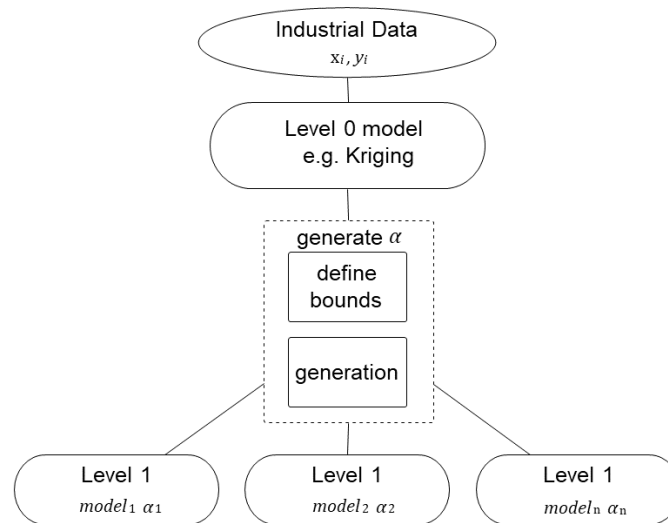


Fig. 1. Simplified process of data-driven test instances generation

2.3. Exploratory Landscape Analysis

One way to extract features of the problem instances is provided by Mersmann, Preuss & Trautmann (2010) by gathering numerical properties from problems with only a few evaluations. The idea was to relate these properties to performance metrics of optimization algorithms for future efficient algorithm selection, e.g., after evaluating a small initial design. Exploratory Landscape Analysis (ELA) characterizes optimization problems by a larger number of numerical feature values grouped into categories (Mersmann et al. 2011).

Numerical computations determine these so-called low-level features. They describe properties like convexity, curvature, local search, and y-distribution, with about 50 numerical sub-features to characterize the problem structure. The low-level features can be computed without expert knowledge, while their relation to so-called high-level features requires expert knowledge. These high-level features describe, e.g., the level of multi-modality, the global structure, the separability, and homogeneity in a somewhat subjective manner and finally result in a classification scheme. Machine Learning (ML) can be used to train algorithm selection models based on benchmark results from the past, e.g., employing data from the platform for Comparing Continuous Optimization algorithms (COCO) on the Black-Box Optimization Benchmark (BBOB, Hansen et al. 2009), using ELA (Bischl et al. 2012, Kerschke & Trautmann 2019a). This paper focuses on ELA features that can be computed using a comprehensive package on the R platform named flacco (Kerschke & Trautmann 2019b).

We are interested in a set of ELA features representing and relating the injection molding optimization problems retrieved by GPS to artificial test problems from the BBOB set to validate the test function generation and calibrate and tune its parameters. BBOB instances and GPS may incorporate randomization of the original problem by translation and scaling, so we follow the procedure used by Škvorc, Eftimov & Korošec (2020) to rely on ELA features that are invariant to these transformations. They developed a generalized method to visualize arbitrary optimization functions to capture their distribution within a benchmark set by implementing a preprocessing procedure that eliminates non-robust and redundant features. The computed ELA features will be visualized two-dimensionally using t-distributed stochastic neighbor embedding (t-SNE, Arora, Hu & Kothari 2018). T-SNE tries to preserve local neighborhood information and tends to cluster neighbors. We use t-SNE as a visual measurement of (dis-) similarity of test instances. We are interested in relations between different data groups, e.g., different types of GPS (e.g., 1stage vs. 2stage, conditional vs. unconditional), compared to a baseline data-set, i.e., optimization problems that do not stem from the IM simulation. Additionally, we aim to analyze which test instance generator setting tends to form recognizable clusters and, therefore, similar test instances. Hopefully, this can be used to detect simulation parameters that dominantly regulate the formation of clusters. Our objective is to find a relevant set of test instances,

with some diversity, as we neither want to test algorithms on instances that are too similar nor too different.

3. EXPERIMENTAL SETUP

We employ a two-step process to evaluate the test instance generation:

1. Numerical comparison of test instances employing ELA, and two-dimensional visualization using t-SNE
2. Analysis of algorithm performances using Elo ratings (Elo 1978)

This section begins with the description of the setup for the first step, which includes the GPS, the BBOB, and the ELA features. It completes with the setup description to compute the Elo rating of competing algorithms on the retrieved test instances.

All experiments are performed using the R platform version 4.1.1. The flacco package (version 1.8.0) was used to compute the ELA features, and the COBBS package (version 1.0.0) to add the GPS instances. To retrieve the BBOB instances, which serve as a baseline comparator, we use the smooF (Single and Multi-Objective Optimization Test Functions, version 1.6.0.2) package.

3.1. Gaussian Process Simulations

We fit the level 0 model, which serves as the ground truth for further comparisons, using Cadmould 3D-F evaluations with three different sizes (10, 15, 20) of Latin hypercube designs (LHD) of the parameter search space (see **Error! Reference source not found.**) and three repetitions each to account for the randomness in the LHDs. The test instances regarding the objectives (y1 to y7) delivered by the IM simulation (see **Error! Reference source not found.**) are fitted using a single GPM each, with the nugget effect disabled, as the IM simulation is deterministic. The model parameter theta (range [1e-6, 1e6]) is optimized using numerical optimization via Differential Evolution (DE, Mullen et al. 2011) with a budget of 200 evaluations per dimension. Three repeats for each design size are performed, leading to 9 ground truth test instances for each objective.

Both GPS variants, the single-stage (1stage) and the two-stage (2stage) variant, are parametrized as follows. A regular grid was used as a design to sample the desired number of test instances (5) for each unique configuration, varying the parameter Ncos from the range [50, 200] with 50 different values and the parameter conditional with the values true and false. This leads to 500 level 1 models (5x50x2) per objective as test instances for the 1stage variant and 500 test instances per objective for the 2stage variant.

3.2. Baseline comparison

To evaluate the retrieved test instances via GPS, we compare them with the artificial test functions from the BBOB test set. We use the function ids (fid) 1-24 and instance ids (iid) 1-10 and set the number of dimensions to three, equivalent to our IM simulation problems, and the parameter search space for each dimension to [-5, 5]. This leads to a total of 240 BBOB test instances.

3.3. ELA feature computation

For each test instance, a set of ELA features will be computed using flacco. Following Škvorec, Eftimov & Korošec (2020), we started with non-correlated features that are also invariant to rotation and shifting. This is useful to a) decrease the dimensionality by focusing on relevant features and b) remove misleading features, as the GPS also may likely shift and rotate the test instance. The resulting list of ELA features after removing redundant features is shown in **Error! Reference source not found.** To compute the selected features for each test instance, flacco was configured to sample them with LHDs of size 1500.

Table 3. Selected uncorrelated landscape features invariant to shift and scale

Feature names 1-7	Feature names 8-14
cm_angle.angle.mean	ela_meta.quad_w_interact.adj_r2
ela_distr.skewness	ela_meta.quad_simple.adj_r2
ela_distr.kurtosis	ela_meta.lin_w_interact.adj_r2
ela_distr.number_of_peaks	disp.ratio_mean_02
ela_meta.lin_simple.adj_r2	disp.ratio_median_25
ela_meta.lin_simple.intercept	pca.expl_var_PC1.cov_init
ela_meta.lin_simple.coef.min	pca.expl_var.cor_init

3.4. Elo rating

Elo (1978) developed his rating system to calculate the relative skill levels of players in chess. Adopted to algorithm performance, we consider each competing algorithm as a player and the test instance as the game. Pairwise comparisons of the algorithm's results after a certain amount of test instance evaluations determine the winning algorithm, i.e., the algorithm that found the lowest objective function value. After every game, the winning player takes points from the losing one, and the number of points is determined by the difference in the two player's ratings. We set the number of repeats for each algorithm on each test instance to 500, i.e., the Elo rating is computed after 500 runs of each algorithm, with an initial Elo score of 1000. The maximum number of function evaluations (i.e., the algorithm budget) is 300. We chose several different algorithms as follows: Generalized Simulated Annealing (GenSA, Xiang et al. 2013), Differential Evolution (DE, Mullen et al. 2011), Bound Optimization BY Quadratic Approximation (BOBYQA, Powell 2009), Constrained Optimization BY Linear Approximations (COBYLA, Powell 1994), and a uniform random search as a baseline comparator. All algorithms with their package, version, and parameter configurations are listed in

GenSA and DE are global optimizers, which have the potential to escape local optima during their search; BOBYQA and COBYLA are local search methods, able to exploit local optima efficiently. We have chosen the Elo score as a robust measurement for the comparison of the strength of single algorithms (compared to the competitors) on single test instances. Our aim is to detect if the Elo scores significantly change after the variation of the simulation parameters. The overall process of generating

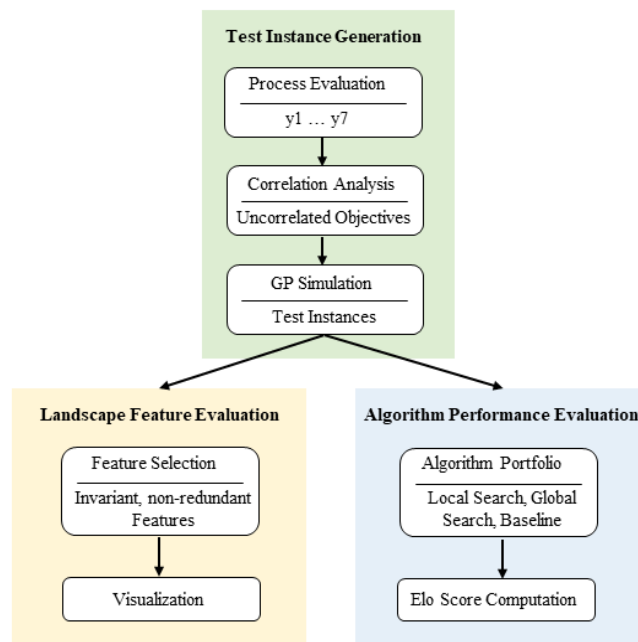


Fig. 2. Overview of the test instance generation and evaluation process

and evaluating the IM simulation-based test instances is depicted in Fig. 2. . For parameters that are not listed, default values were applied.

Table 4. Chosen single objective optimizers with package, version, and parameter setting

Algorithm	Package (version)	Parameter	Value
GenSA	GenSA (1.1.7)	acceptance_param	-5
		visiting_param	2.62
		temperature	5230
DE	DEoptim (2.2-6)	NP	20
		F	0.8
		CR	0.9
		strategy	2
BOBYQA	minqa (1.2.4)	-	-
COBYLA	nloptr (1.2.2.2)	-	-
RS	-	-	-

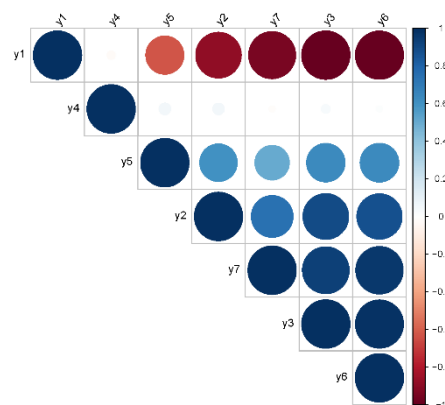


Fig. 3. Pairwise correlations between the objectives in an ascending order from top to bottom

GenSA and DE are global optimizers, which have the potential to escape local optima during their search; BOBYQA and COBYLA are local search methods, able to exploit local optima efficiently. We have chosen the Elo score as a robust measurement for the comparison of the strength of single algorithms (compared to the competitors) on single test instances. Our aim is to detect if the Elo scores significantly change after the variation of the simulation parameters. The overall process of generating and evaluating the IM simulation-based test instances is depicted in Fig. 2.

4. RESULTS

4.1. Landscape Feature Visualization using t-SNE

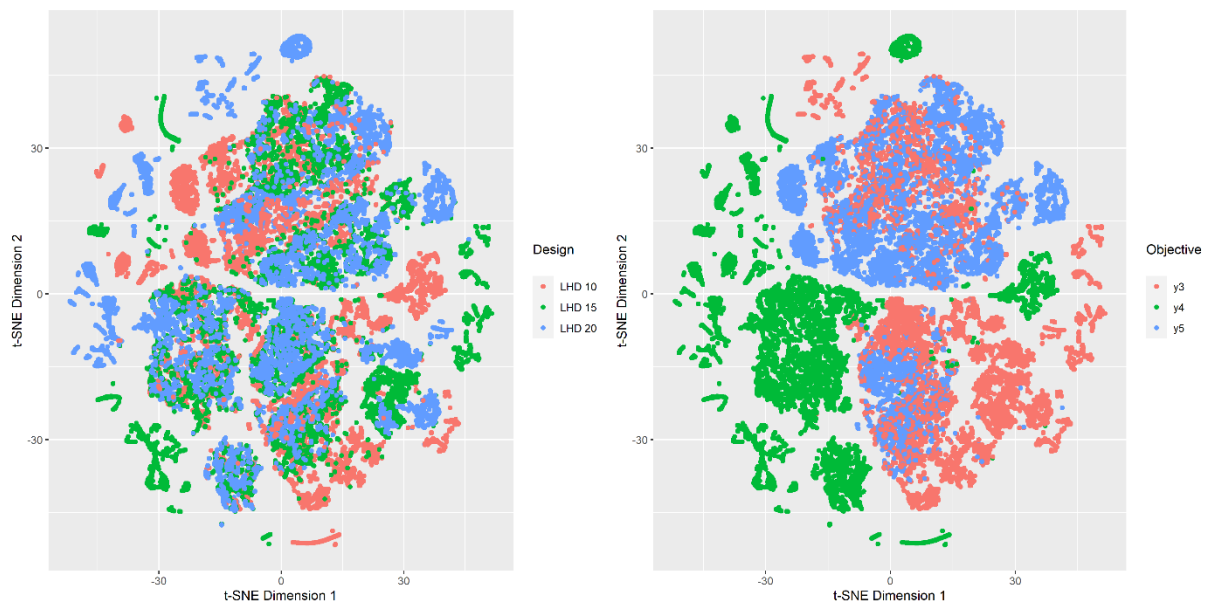


Fig. 4. T-SNE visualization of the feature space of all computed test instances based on different design sizes (10, 15, 20) and selected objectives (y3, y4, y5). The left plot is colored based on the design size, and the right plot based on the objectives.

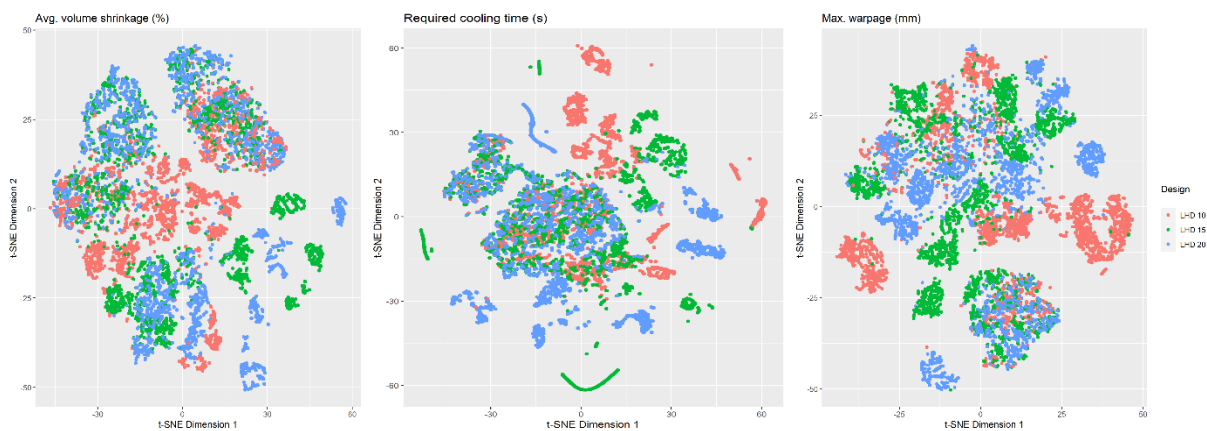


Fig. 5. t-SNE visualization for each objective with respect to the design size

First, we computed all objectives' correlation coefficients using the IM simulations' LHD samples. The correlation matrix is shown in Fig. 3. The first objective, y1, the element-related mass, is highly negatively correlated to the objectives y2, y3, y6, and y7. This is intuitively explainable, as finding a parameter setup that maximizes the mass to the desired value will minimize objectives like shrinkage and deformation and vice versa. The objectives y4 and y5 are not highly correlated to other objectives. Additionally, the objectives y2, y3, y6, and y7 are highly positively correlated. As a consequence, we choose y3 as a representative objective for them. Objective y4, the required cooling time, addresses the process's cycle time. This results in the following uncorrelated objectives to minimize, which will be analyzed further: y3 (avg volume shrinkage), y4 (required cooling time), and y5 (max. warpage).

The first parameter of the test instance generation to be analyzed visually is the design size. For a first overview, all simulation instances (1stage and 2stage) of the selected three objectives (y3-y5) are chosen, and the computed features are visualized in two dimensions using t-SNE (see Fig. 4). The

different objectives tend to form more recognizable clusters than the design size, which shows more overlapping, especially in the center of the plot.

Deeper insight can be taken from Fig. 5, where the instances are visualized per objective. At first glance, all objectives have more overlaps on the design sizes 15 and 20, while there are more single clusters of design size 10-based test instances recognizable. Therefore, design size 10 might not generate test instances with the same characteristics as sizes 15 and 20. Further analysis regarding the minimum design size to use will be approached based on the performances of algorithms, which will be presented in Sec. 4.2.

To reduce the complexity of the remaining visualizations, we will focus on the LHDs with the largest design size, 20, which are likely the most accurate. Fig. 6 depicts the instances per objective w. r. t. the method, i.e., 1stage-simulations, 2stage-simulations, Groundtruth (the estimate of the GPM trained on the Cadmould simulation), or BBOB. First, the BBOB instances are forming separate clusters, which indicates that, according to the selected ELA features, the IM test instances differ from the artificial BBOB instances.

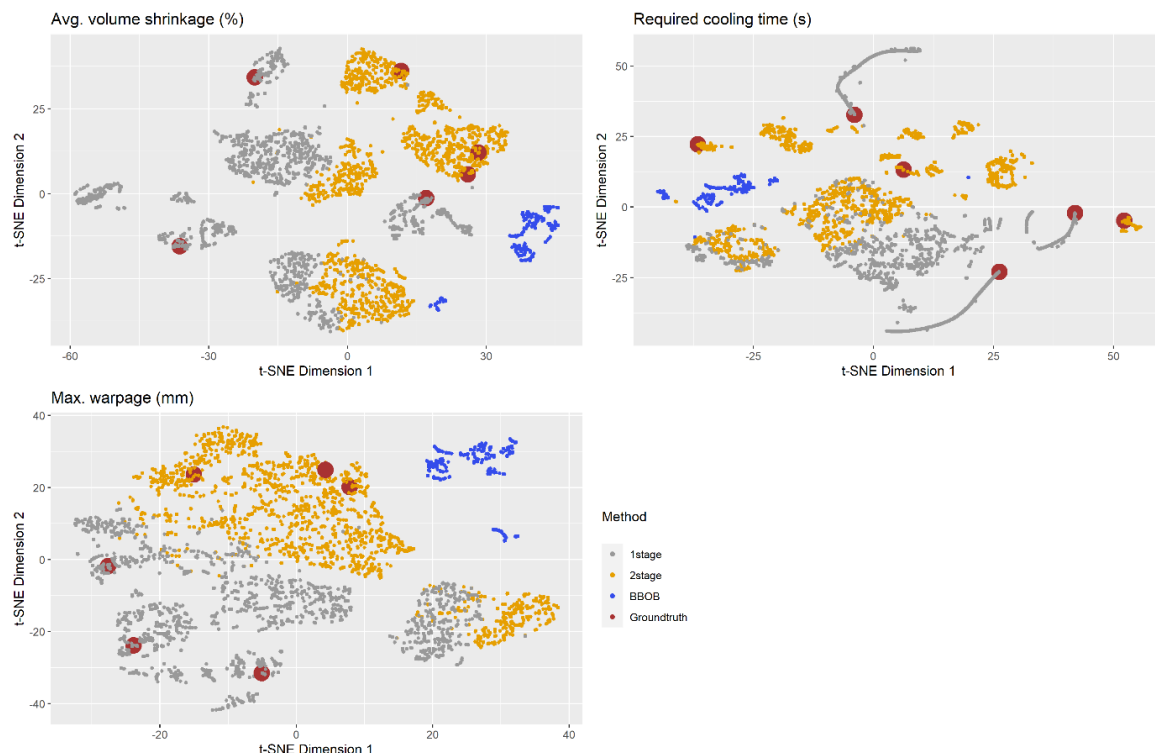


Fig. 6. T-SNE visualization for each objective with respect to the test instance generation method. Simulations and Groundtruth are computed using the LHD 20. Please note that the six Groundtruth points are significantly increased in size to be recognizable.

Interestingly, there is a recognizable difference between the 1stage and 2stage simulations, which likely means they do not produce very similar test instances. Additionally, the objective max. warpage seems more space-filling compared to the others. Consequently, we will take a deeper look at the objective max. warpage w. r. t. the simulation method and the parameters Ncos and conditional.

These results are visualized in Fig. 7. Both the simulation method and the conditioning to the training data play a significant role in distinguishing test instances of the selected objective, while the parameter Ncos does not seem to form obvious clusters.

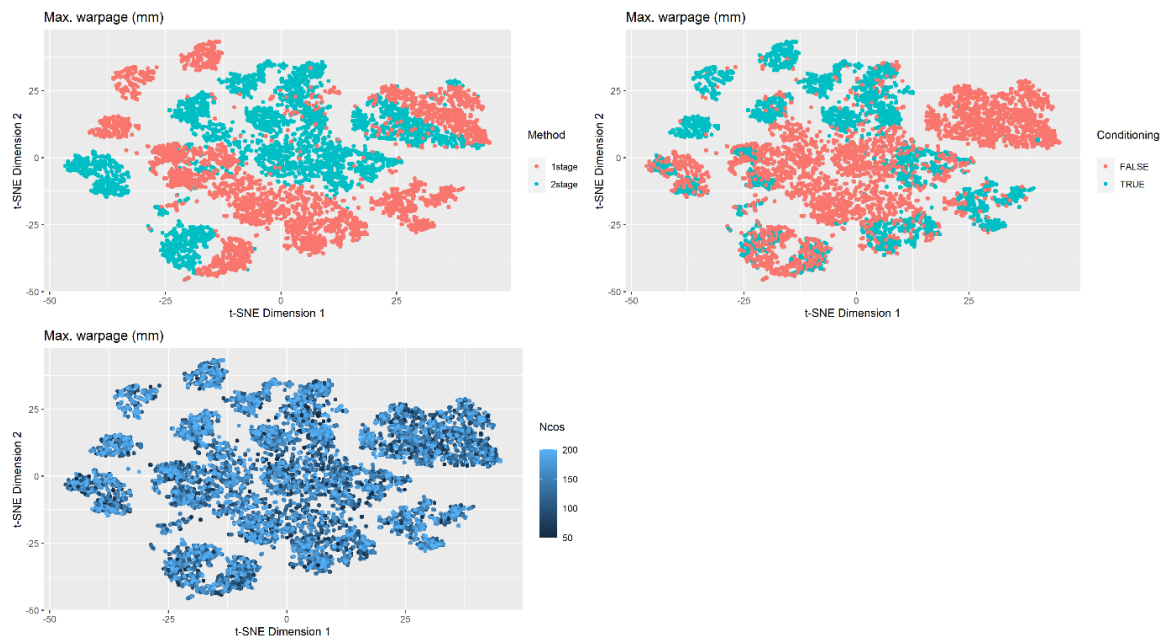


Fig. 7. T-SNE visualization of the objective max. warpage [mm] (y5) with respect to the simulation method, the conditioning, and the number of cosines.

The benchmark experiment results in the following subsection will examine if there is also a difference in the resulting ranks of the algorithm's performance between 1stage and 2stage instances to conclude if and where 2stage simulations are helpful or necessary.

4.2. Performance Evaluation using Elo Scores

In this section, we analyze the relative performance of the selected algorithms on one randomly selected test instance per configuration, i.e., the design size (10, 15, 20), the simulation method (1stage, 2stage), the conditioning (TRUE, FALSE), the Ncos (50-200), and the objective (y3-y5). We are especially interested in changes in the algorithm's ranking on the objectives by varying simulation parameters, i.e., answering the question if the simulation parameters are changing the difficulty for some of the algorithms.

The Elo scores are depicted in Fig. 8. Either GenSA or BOBYQA shows superior performance, i.e., a high Elo score, depending on the objective or simulation configuration. It can be assumed that BOBYQA performs well on linear or quadratic, i.e., unimodal, optimization problems. This can be observed especially on objective y4 (required cooling time). Additionally, COBYLA also performs generally well on y4, underlying the linear character of the problem. On y4, not many significant changes in performance rankings related to the simulation parameter are recognizable. COBYLA seems to suffer on unconditional simulations. Additionally, for y4, the design size does not significantly influence the performance rankings, and the competing algorithms consistently outperform RS. On objective y3 (avg. volume shrinkage), GenSA shows high relative performance. Reading the cells from left to right indicates that RS increases its Elo score while BOBYQAs relative performance decreases. Especially for LHD 20, 2stage, and unconditional simulations, the Elo scores of all algorithms are relatively close, indicating the difficulty of this setup for this objective. For objective y5 (max. warpage), this can also be observed, contrarily to objective y4, where 2stage simulations and larger values for Ncos are not significantly changing the overall pattern of the Elo scores.



Fig. 8. Elo-scores of all selected algorithms on one randomly selected test instance per configuration shown on the row and column labels. A higher Elo value indicate a better relative performance.

A closer look at changes in Elo scores of BOBYQA and DE are given in Fig. 9 and Fig. 10 resp. The cells are reordered such that 1stage and 2stage simulations are visualized next to each other, and to the right, the value of Ncos increases. This reveals the stronger effect of the simulation method and the performance advantage of the GenSA algorithm on the objectives y3 and y5.

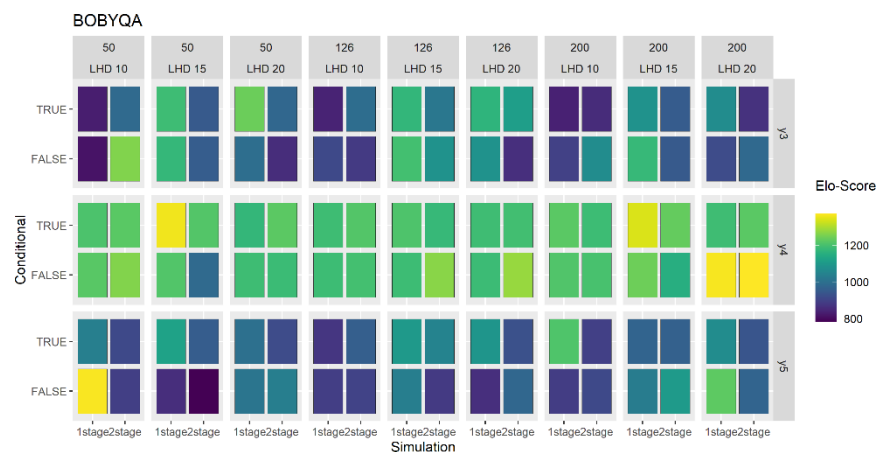


Fig. 9. Elo scores of the BOBYQA algorithm

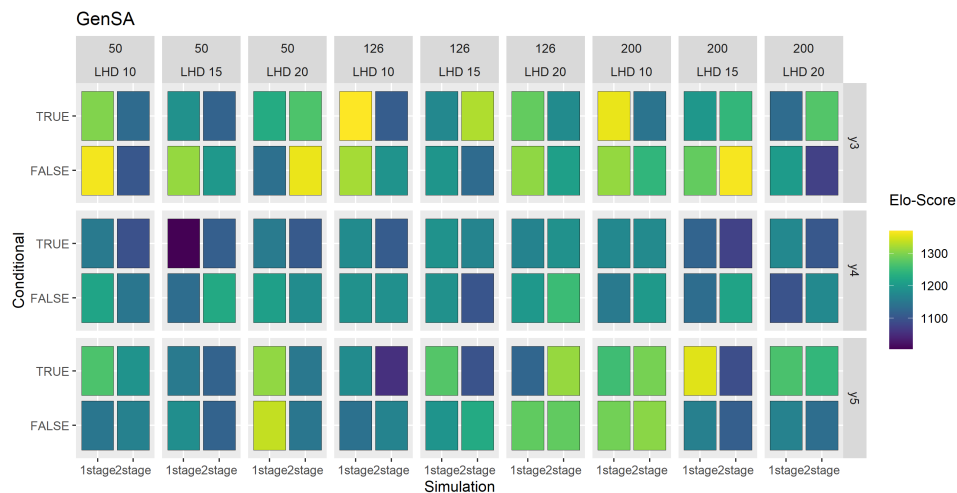


Fig. 10. Elo scores of the GenSA algorithm

To conclude, the simulation parameters have a higher effect on more complex objectives, like y3 and y5, than simpler objectives like y4. We opt for small size LHD sampling and the most simplistic GPS for y4 and at least LHD 15 for the other objectives. The 2stage simulation method combined with a higher number of Ncos seems to create the optimization problems with the highest difficulty, which can be helpful depending on the optimization goal and the hardness of the underlying industrial process to optimize. The conditioning has a significant effect on the relative performance. Simulation instances conditioned to the training data stemming directly from the process to optimize can be beneficial, when they are additionally used for regression or predictions, e.g., employed by sequential parameter optimization. Otherwise, unconditional simulation can be used for robust algorithm selection and parameter tuning, reflecting likely variations of the process. For a more detailed suggestion, further experiments are needed. As each configuration was only examined using the Elo rating on a single, randomly drawn, test instance, an extension to and aggregation of the results on several instances is necessary.

5. DISCUSSION

Numerical optimization requires the selection of appropriate and efficient optimization algorithms. It is beneficial to exploit all available information about the problem to support this task. Benchmarks are especially useful for successful optimization, but relevant test problems are often unavailable, whereas some process data are available. We exploit this data for data-driven test instance generation and validate the generation process by comparing retrieved instances with the ground truth and an artificial test set (BBOB). Furthermore, we analyzed several sizes of the initial training data and the parameterization of the GPS, i.e., 1stage or 2stage simulation, the conditioning, and the parameter Ncos.

Visual representation of selected ELA features reduced to two dimensions using t-SNE reveals the neighborhood information based on the simulation parameters and the objectives. Combining the landscape feature analysis with the relative performance of selected algorithms of different algorithm families, i.e., global and local search heuristics, allows for the detection of patterns in the test set like softening or hardening the difficulty of the test set.

Please note that the visualization of the feature space changes if other features are added. Additionally, applying dimensionality reduction techniques introduces the loss of some information. Using a relative performance measure of the selected algorithms introduces some bias, i.e., the addition of low-performing algorithms (or other algorithms in general) will change the Elo scores.

6. CONCLUSIONS

This article shows how ELA features can be used to visualize the problem space of injection molding optimization problems according to different objectives by applying Gaussian process simulation. We analyzed how ELA-based visualization can be applied to a real-world problem, comparing several problem classes in the feature space and how they relate to traditional artificial benchmark sets like the BBOB. A second step analyzed the performance of a set of different algorithms using the Elo rating as a measurement of relative performance. Regarding the research questions in the focus of this paper, we draw the following conclusions:

(RQ-1) Are GPMs leading to relevant test instances for the IM simulation?

From our point of view, regression and interpolation models are a good choice for filling the gap of process-related test instance, as artificial test instance sets like BBOB are not able to cover all optimization problems. It can be seen that the GPS instances show a smaller variance regarding the ELA features within a problem class (e.g., for a single objective using a fixed configuration of GPS) compared to, e.g., the BBOB instances. The t-SNE visualization reflects this. In the next step, we will experimentally analyze the efficiency of parameter tuning on both the GPS and BBOB test instances for the IM optimization.

(RQ-2) What is the necessary (minimum) number of evaluations of the IM simulation to train the test instance generator for meaningful instances?

A low number of process samples are valid for simplistic (e.g., unimodal) problems (e.g., 10 evaluations). Giving a fixed number for all objectives is not feasible at this point. More complex objectives might be smoothened too much by the Gaussian process models, i.e., the characteristic of the underlying problem might be changed if the search space is sampled too sparsely. We recommend a conservative approach, e.g., at least 15 samples of the non-linear and multimodal objectives for the IM simulation to capture more details of the process.

(RQ-3) How do the IM test instances under investigation behave w.r.t. their type and parametrization, i.e., 1stage or 2stage, the number of cosines configured for spectral simulation, and the conditioning to the training data?

First, no single parameter dominates either the ELA-based clustering of the GPS instances or the relative performance based on the Elo-scores of the algorithms. Complex multimodal problems, like the max. warpage, might be more sensitive to a combined change, e.g., unconditioned 2stage simulation with a larger value for Ncos, which influences the performance of simple local search heuristics like BOBYQA or even RS to the negative. This indicates that, depending on the experimenter's goal, more complex simulations can harden the test instances and therefore deliver a more robust benchmark. This will be further analyzed in future work, i.e., with experiments based on a larger number of instances, allowing for a more generalizable result, and validated on an injection molding machine. It can be seen that the parameter Ncos does not have a huge impact on both, the ELA-based feature space visualization and the Elo-based performance evaluation.

This paper demonstrates the process of combining ELA-based comparisons of test instances with Elo rating-based performance measure on the example of Injection molding optimization. Further research will examine how the applied simulation methods can be used for efficient optimization. The applied procedure of data-driven test function generation allows for rigorous benchmarking experiments with the goal of, e.g., algorithm selection and hyperparameter optimization and, therefore, efficient optimization of the underlying process. It can be seen that there is not a single optimizer superior performing on the different objectives. To derive benchmark-based conclusions for the whole problem class, i.e., a set of optimization problem instances, at least a subset of sample instances is needed. We exemplified the test instance generation and validation procedure on the example of IM simulations, but it might as well be extended and adopted to several (black-box) optimization processes. The results show both the need for and the effects of calibration and tuning of the simulation parameters.

ACKNOWLEDGMENTS

The presented work is funded by the European Union from the European Regional Development Fund (EFRE) and the State of North Rhine-Westphalia, Germany. (EFRE- 0500131)

REFERENCES

1. Arora, S, Hu, W & Kothari, PK 2018, “An analysis of the t-sne algorithm for data visualization”, *Proceedings of the 31st Conference On Learning Theory*, PMLR, pp. 1455-1462.
2. Bischl, B, Mersmann, O, Trautmann, H & Preuss, M 2012, “Algorithm selection based on exploratory landscape analysis and cost-sensitive learning”, *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, New York, NY, USA, pp. 313-320.
3. Elo, AE 1978, *The Rating of Chess Players, Past and Present*, Arco Pub., New York.
4. Fischbach, A, Zaefferer, M, Stork, J, Friese, M & Bartz-Beielstein, T 2016, “From real world data to test functions”, *Proceedings 26. Workshop Computational Intelligence*, KIT Scientific Publishing, pp. 159-177.
5. Forrester, A, Sobester, A & Keane, AJ 2008, *Engineering Design via Surrogate Modelling: A Practical Guide*, John Wiley & Sons, Ltd.
6. Hansen, N, Auger, A, Finck, S, & Ros, R 2009, “Real-parameter black-box optimization benchmarking: Experimental setup”, Technical Report RR-6828, INRIA, viewed 28 Feb. 2022, <https://hal.inria.fr/inria-00362649v3/document>
7. Kerschke, P & Trautmann, H 2019a, “Automated Algorithm Selection on Continuous Black-Box Problems by Combining Exploratory Landscape Analysis and Machine Learning”, *Evolutionary Computation*, vol. 27, no. 1, pp. 99–127.
8. Kerschke, P & Trautmann, H 2019b, “Comprehensive Feature-Based Landscape Analysis of Continuous and Constrained Optimization Problems Using the R-package flacco”, in N Bauer, K Ickstadt, K Lübke, G Szepannek, H Trautmann & M Vichi (ed.), *Studies in Classification, Data Analysis, and Knowledge Organization*, Springer, Cham, pp. 93-123.
9. Mersmann, O, Preuss, M & Trautmann, H 2010, “Benchmarking Evolutionary Algorithms: Towards Exploratory Landscape Analysis“, *Parallel Problem Solving from Nature - PPSN XI*, Springer, Berlin, Heidelberg, pp. 73-82.
10. Mersmann, O, Bischl, B, Trautmann, H, Preuss, M, Weihs, C & Rudolph, G 2011, “Exploratory landscape analysis”, *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, New York, NY, USA, pp. 829-836.
11. Mullen, K, Ardia, D, Gil, D, Windover, D & Cline, J 2011, “DEoptim: An R Package for Global Optimization by Differential Evolution”, *Journal of Statistical Software*, vol. 40, no. 6, pp.1-26.
12. Powell, MJD 1994, “A direct search optimization method that models the objective and constraint functions by linear interpolation”, In: Gomez, S., Hennart, JP. (eds) *Advances in Optimization and Numerical Analysis. Mathematics and Its Applications*, vol 275. Springer, Dordrecht.
13. Powell, MJD 2009, “The BOBYQA algorithm for bound constrained optimization without derivatives”, Technical Report No. DAMTP 2009/NA06, Centre for Mathematical Sciences, University of Cambridge, UK, viewed 28 Feb. 2022, http://www.damtp.cam.ac.uk/user/na/NA_papers/NA2009_06.pdf
14. Škvorec, U, Eftimov, T & Korošec, P 2020, “Understanding the problem space in single-objective numerical optimization using exploratory landscape analysis”, *Applied Soft Computing*, vol. 90, no. 106138.

15. Xiang, Y, Gubian, S, Suomela, B & Hoeng, J 2013, “Generalized Simulated Annealing for Global Optimization: The GenSA Package”, *The R Journal*, vol. 5, no.1, pp. 16-28.
16. Zaefferer, M, Fischbach, A, Naujoks, B & Bartz-Beielstein, T 2017, “Simulation-based test functions for optimization algorithms”, *Genetic and Evolutionary Computation Conference 2017*, ACM, Berlin, pp. 905-912.
17. Zaefferer, M & Rehbach, F 2020, “Continuous Optimization Benchmarks by Simulation”, *Parallel Problem Solving from Nature - PPSN XVI*, Springer, Cham, pp. 273-286.
18. Zaefferer, M 2020, *COBBS: Continuous Optimization Benchmarks By Simulation*, R-Package available on Github, viewed 28. Feb. 2022, <https://github.com/martinzaefferer/COBBS>