Exploratory Landscape Analysis

Olaf Mersmann[*]
TU Dortmund, Germany
olafm@statistik.tu-dortmund.de

Bernd Bischl
TU Dortmund, Germany
bischl@statistik.tu-dortmund.de

Heike Trautmann
TU Dortmund, Germany
trautmann@statistik.tu-dortmund.de

Mike Preuss
TU Dortmund, Germany
mike.preuss@tu-dortmund.de

Claus Weihs
TU Dortmund, Germany
weihs@statistik.tu-dortmund.de

Günter Rudolph
TU Dortmund, Germany
guenter.rudolph@tu-dortmund.de

ABSTRACT

Exploratory Landscape Analysis (ELA) subsumes a number of techniques employed to obtain knowledge about the properties of an unknown optimization problem, especially insofar as these properties are important for the performance of optimization algorithms. Where in a first attempt, one could rely on high-level features designed by experts, we approach the problem from a different angle here, namely by using relatively cheap low-level computer generated features. Interestingly, very few features are needed to separate the BBOB problem groups and also for relating a problem to high-level, expert designed features, paving the way for automatic algorithm selection.

Categories and Subject Descriptors

G.1.6 [**Optimization**]: Global Optimization, Unconstrained Optimization; G.3 [**Probability and Statistics**]: Statistical Computing; I.2.6 [**Learning**]: Knowledge Acquisition

General Terms

Experimentation, Algorithms, Performance

Keywords

evolutionary optimization, fitness landscape, exploratory landscape analysis, BBOB test set, benchmarking

1. INTRODUCTION

The development of evolutionary and related optimization methods for real-valued problems is still a dynamic research area as it sees a large number of newly designed algorithms each year. These are usually evaluated on sets of benchmark problems such as the CEC'05 (Suganthan et al., 2005) and the BBOB'09/'10 sets (Hansen et al., 2009), and conclusions are drawn based on majorities (algorithm A outperforms algorithm B on more problems and vice versa) or via consensus rankings (Mersmann et al. (2010)). However, in many cases, not much effort is put in detecting the strengths and weaknesses of such algorithms in terms of problem properties, with the notable exception of dimensionality. The CEC'05 benchmark and to an even larger extent the BBOB benchmark have been designed to enable exactly that: To find out which algorithm is good on which *problem group*. Meanwhile, the naive belief in a single optimization algorithm that dominates all others even for a certain domain (e.g. multi-modal problems) has lost ground and the evidence gathered from experimental analysis leads to another approach: Given that we possess a good overview over the properties that make a problem easy or hard for a specific algorithm, we may choose the 'right' algorithm from an ensemble to solve it efficiently.

However, since knowledge about the interaction of problem properties and algorithm efficiency is incomplete even for constructed benchmark problems, it is even more difficult to choose an algorithm for a real-world problem since for these even less is known about the function. In order to find the 'right' optimization algorithm for such an optimization problem, we thus need two things: Improved knowledge about problem properties which capture the difficulty of an optimization problem for a specific algorithm, and a method to estimate these properties relatively cheaply.

In this work, we concentrate on the second task, following the path termed *exploratory landscape analysis* (ELA) as suggested in Mersmann et al. (2010) and Preuss and Bartz-Beielstein (2011). In addition to the expert designed features of Mersmann et al. (2010), we suggest a number of low-level features of functions that can estimated from relativly few function evaluations. Both feature groups are introduced in section 2. The latter is first used to predict to which of the five BBOB problem groups each of the 24 BBOB test functions belongs. In order to keep the number of features or rather the number of necessary function evaluations as low as possible, we employ a multi-objective *evolutionary algorithm* (EA) to simultaneously optimize feature sets according to quality and cost, with exciting results as presented in section 4. The required machine learning methods are described briefly in section 3.

In a second step, we try to relate the low-level features to the expert designed high-level features of Mersmann et al. (2010), which poses a much more demanding task. Again, a

[*]Corresponding author

multi-objective EA is used to determine sets of good and not too expensive features, as demonstrated in section 5. Our reasoning is that if a human expert is able to select or design an algorithm suitable for a fixed function given the high-level features of that function, and if the high-level features can be predicted by evaluating the function a few times for well chosen parameter settings to estimate the value of the low-level features, then it should be possible to automatically select a good algorithm for a fixed function. However, we have yet to achieve the last step and therefore it is more of a vision to be worked on further in upcoming works.

To obtain a better idea of the usefulness of the low-level features, we lastly compare the feature sets resulting from the two experiments in section 5.2 before concluding.

2. EXPLORATORY LANDSCAPE ANALYSIS

Exploratory Landscape Analysis originates from the observation that once a problem is well known, one can employ a matching optimization algorithm to solve it. However, most problems encountered in practice (e.g. from the engineering domain) are poorly understood. It is a common practice to either run some initial tests with one or more algorithms and then adapt the algorithm to the problem according to the obtained results, or to apply a 'blind' parameter tuning process to do the same automatically. If computing one fitness evaluation is costly, both approaches are problematic. This often results in insufficient algorithm comparisons in practice. We follow a different reasoning by finding interactions between problem properties and algorithms so that the problem features defining a specific algorithm's performance can be gathered without actually running it. Instead, these properties are estimated using a small sample of function values combined with statistical and machine learning techniques. These provide us with insights into the nature of the problem and enable us to select a good candidate optimization algorithm.

This automated knowledge acquisition approach of course heavily depends on a good feature set. We therefore start with a short summary of already known expert designed high-level features, followed by suggesting a new low-level feature set which constitutes the basis of the experimental investigation in this work.

2.1 Properties based on Expert Knowledge

Mersmann et al. (2010) discuss eight properties to characterize the complexity of an optimization problem which can be specified based on expert knowledge of the problem at hand. However, other interpretations are possible and we do not assume incontrovertible general acceptance. These properties are used to characterize the Black-Box Optimization Benchmarking (Hansen et al., 2009, BBOB) test functions (see Table 1) and are then compared and contrasted to the performance of the competing optimization algorithms in the contest.

In addition to the self-explanatory aspect of **multi-modality**, the existence of a **global structure** is an indicator for the challenges of a given optimization problem. A lack of global basin structure severely increases the search effort of an algorithm. This holds as well for non-separable problems which cannot be partitioned into lower-dimensional sub-problems. Thus, problem **separability** is an important property as well. **Variable scaling** may lead to non-spherical basins of attraction which require a different behavior of the algorithm in each dimension especially with focus on algorithm step size.

Search space- and **basin size homogeneity** both relate to the overall appearance of the problem. Problem hardness is surely dependent on the existence of phase transitions in the search space as well as on the variation of the sizes of all basins of attractions. On the contrary, a high **global to local optima contrast**, which refers to the difference between global and local peaks in comparison to the average fitness level, facilitates the recognition of excellent peaks in the objective space. The feature characterizing the function landscapes plateaus is omitted in this study because the test function set lacks variation w.r.t. this feature.

Function	multim.	gl.-struc.	separ.	scaling	homog.	basins	gl.-loc.
1 Sphere	none	none	high	none	high	none	none
2 Ellipsoidal separable	none	none	high	high	high	none	none
3 Rastrigin separable	high	strong	none	low	high	low	low
4 Bueche-Rastrigin	high	strong	high	low	high	med.	low
5 Linear Slope	none	none	high	none	high	none	none
6 Attractive Sector	none	none	high	low	med.	none	none
7 Step Ellipsoidal	none	none	high	low	high	none	none
8 Rosenbrock	low	none	none	none	med.	low	low
9 Rosenbrock rotated	low	none	none	none	med.	low	low
10 Ellipsoidal high-cond.	none	none	none	high	high	none	none
11 Discus	none	none	none	high	high	none	none
12 Bent Cigar	none	none	none	high	high	none	none
13 Sharp Ridge	none	none	none	low	med.	none	none
14 Different Powers	none	none	none	low	med.	none	none
15 Rastrigin multi-modal	high	strong	none	low	high	low	low
16 Weierstrass	high	med.	none	med.	high	med.	low
17 Schaffer F7	high	med.	none	low	med.	med.	high
18 Schaffer F7 mod. ill-cond.	high	med.	none	high	med.	med.	high
19 Griewank-Rosenbrock	high	strong	none	none	high	low	low
20 Schwefel	med.	deceptive	none	none	high	low	low
21 Gallagher 101 Peaks	med.	none	none	med.	high	med.	low
22 Gallagher 21 Peaks	low	none	none	med.	high	med.	med.
23 Katsuura	high	none	none	none	high	low	low
24 Lunacek bi-Rastrigin	high	weak	none	low	high	low	low

Table 1: Classification of the noiseless BBOB functions based on their properties (multi-modality, global structure, separability, variable scaling, homogeneity, basin-sizes, global to local contrast). Predefined groups are separated by horizontal lines.

2.2 Low-Level Features

We consider six low-level feature classes in this work to characterize the structure of an unknown fitness landscape. The starting point is a data set of s different decision variable settings X^s with respective objective values Y^s generated by a random Latin hypercube (LH) design covering the decision space. We will denote the combination of parameter settings and corresponding function values by $D^s = [X^s, Y^s]$. The size of the LH desgin increases linearly with the dimension d of the problem in order to account for the increasing problem complexity, i.e. the number of initial points is chosen as $s = c \cdot d$, where c is a predefined constant.

The basic concepts of the feature classes are described in the following. It has to be noted that each class contains several lower level sub-features which, in each case, can be generated using the same experimental data pool. All of these 50 sub-features are only briefly summarized here, due to lack of space (see Table 2 for details)[1]

Convexity: Two random points from X^s are selected and a linear combination with random weights is formed. The

[1]R source code to calculate the features for a given function can be obtained from `http://ptr.p-value.net/gco10/ela.R`. Additional plots and the raw data generated by these experiments is available from `http://ptr.p-value.net/gco10/suppl.zip`.

difference of the objective value of the new point and the convex combination of the objective values of the two sampled points is computed. This is repeated 1000 times, and the average number of instances where the calculated difference is less than a predefined negative threshold (here -10^{-10}) related to the number of repetitions estimates the probability of convexity. Linearity of the function is assumed in case all absolute differences are smaller than the absolute value of the specified threshold.

y - Distribution: The distribution of Y^s is characterized by the skewness and kurtosis of the objective function values. The latter measures the degree of peakedness of the distribution, i.e. reflects whether the distribution is rather flat or peaked when compared to a normal distribution. The number of peaks of the distribution is estimated as well as an indicator for multi-modality.

Levelset: The initial data set D^s is split into two classes by a specific objective level which works as a threshold. One possibility is to use the median for this, which will result in equally sized classes. Other choices studied are the upper and lower quartiles of the distribution of y. Linear (LDA), quadratic (QDA) and mixture discriminant analysis (MDA) are used to predict whether the objective values Y^s fall below or exceed the calculated threshold. Multi-modal functions should result in several unconnected sublevel sets for the quantile of lower values, which can only be modeled by MDA, but not LDA or QDA. The extracted low-level features are based on the distribution of the resulting cross-validated mean misclassification errors of each classifier.

Meta-Model: Linear and quadratic regression models with or without interactions are fitted to the initial data D^s. The adjusted coefficient of determination R^2 is returned in each case as an indicator for model accuracy. Functions with variable scaling will not allow a good fit of regression models without interaction effects, and simple unimodal functions might be approximated by using a quadratic model. In addition, features are extracted which reflect the size relations of the model coefficients.

Local search: A local search algorithm (Nelder-Mead) is started from a random sample of size $N = 50 \cdot d$ from D^s. The solutions are hierarchically clustered in order to identify the local optima of the function. The basin size is approximated by the number of local searches which terminate in each identified local optimum. Extracted features are the number of identified clusters as an indicator for multi-modality as well as characteristics related to the basin sizes around the identified local optima reflected by the sizes of the particular clusters. In addition, summary statistics like the extrema and specific quartiles of the distribution of the number of function evaluations spent by the local search algorithm runs until termination are computed.

Curvature: For $100 \cdot d$ points from D^s, the gradient is numerically approximated in each point using Richardson's extrapolation method (Linfield and Penny (1989)). Resulting features are summary statistics of the Euclidean norm of the latter as well as of the relations of maximum and minimum values of the respective partial derivatives. Furthermore, the same statistics are applied to the condition number of the numerical approximation of the Hessian (Linfield and Penny (1989)).

As visualized in Fig. 1 the low-level feature classes cover different aspects of the defined high-level features. To empirically justify these relationships, the high-level features are

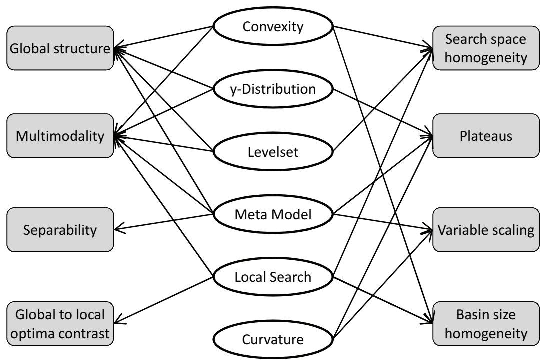

Figure 1: Relationships between high-level features (grey) and low-level feature classes (white)

compared to the computed low-level features on the BBOB test functions. Statistical classification techniques are used to predict the level of each high-level feature (see Table 1), e.g. low, medium or high multi-modality, from the values of the low-level features. Representative results are presented in section 5.

In addition, in section 4, the five predefined function groups of the BBOB contest (1. separable, 2. low or moderate conditioned, 3. high conditioned and unimodal, 4. multi-modal with adequate global structure, 5. multi-modal with weak global structure) as listed in Hansen et al. (2009) are predicted based on the low-level features.

3. MACHINE LEARNING METHODS

Modeling: In order to solve the various classification tasks posed in this article we will always use a random forest as proposed by Breiman (2001). This model creates an ensemble of unpruned classification trees based on bootstrapped versions of the training data. Predictions of the individual trees are aggregated by majority voting. As the following experiments will be complex and time-consuming enough, we do not want to burden ourselves with model selection and therefore choose a classifier which is reliable in the following sense: it can capture nonlinear relationships between features and classes and is known to exhibit strong performance in many applications. Although it might be tunable for specific problems, its performance usually does not degrade drastically if the settings of its hyperparameters are not optimal. This is in strong contrast to other competitive classification methods, e.g. SVMs. Finally, it is invariant under all monotone transformations of the features and therefore different or irregular scalings constitute no problem for the forest.

Performance estimation: Although the misclassification error of the forest can usually be assessed in a computationally cheap and statistically unbiased way by using its internal out-of-bag bootstrap estimator, we opt here for a different approach. Since for each instance of a BBOB test function five identical replications exist, we want to avoid that the classifier simply memorizes the functions it has already seen. We therefore choose a specific 5-fold cross-validation strategy, which forces the classifier to generalize to function instances not seen in the training set. This is ensured by either having all replications of a parametrized function completely in the training or test set (stated differently we "cross-validate on the BBOB function instances"). Later on we study an even harder setting by calculating the performance of the classifier

Feature group and name	Description
Meta-model features:	
1 `approx.{linear,lineari}_ar2`	adjusted R^2 of the estimated linear regression model without and with interaction
1 `approx.linear_{min,max}_coef`	minimum and maximum value of the absolute values of the linear model coefficients
2 `approx.{quadratic,quadratici}_ar2`	adjusted R^2 of the estimated quadratic regression model without or with interaction
2 `approx.quadratic_cond`	maximum absolute value divided by minimum absolute value of the coefficients of the quadratic terms in the quadratic model
Convexity features:	
3 `convex.{linear,convex}_p`	estimated probability of linearity and convexity
3 `convex.linear_dev`	mean deviation from linearity
y distribution features:	
4 `distr.skewness_y`	skewness of the distribution of the function values
4 `distr.kurtosis_y`	kurtosis of the distribution of the function values
4 `distr.n_peaks`	estimation of the number of peaks in the distribution of the function values
Levelset features:	
5 `levelset.lda_mmce_{10,25,50}`	mean LDA misclassification error for function values split by 0.1, 0.25, 0.5 quantile (estimated by CV)
5 `levelset.lda_vs_qda_{10,25,50}`	`levelset.lda_mmce_{10,25,50}` divided by `levelset.qda_mmce_{10,25,50}`
6 `levelset.qda_mmce_{10,25,50}`	mean QDA misclassification error for function values split by 0.1, 0.25, 0.5 quantile (estimated by CV)
7 `levelset.mda_mmce_{10,25,50}`	mean MDA misclassification error for function values split by 0.1, 0.25, 0.5 quantile (estimated by CV)
Local search features:	
8 `ls.n_local_optima`	number of local optima estimated by the number of identified clusters
8 `ls.best_to_mean_contrast`	minimum value of cluster centers divided by the mean value of cluster centers
8 `ls.{best,worst}_basin_size`	proportion of points in the best and worst cluster
8 `ls.mean_other_basin_size`	mean proportion of points in all clusters but the cluster with the best cluster center
8 `ls.{min,lq,med,uq,max}_feval`	0, 0.25, 0.5, 0.75 and 1 quantile of the distribution of the number of function evaluations performed during a single local search
Curvature features:	
9 `numderiv.grad_norm_{min,lq,med,uq,max}`	minimum, lower quantile, median, upper quantile and maximum of the euclidean norm of the estimated numerical gradient
9 `numderiv.grad_scale_{min,lq,med,uq,max}`	minimum, lower quantile, median, upper quantile and maximum of the maximum divided by the minimum of the absolute values of the estimated partial gradients
10 `numderiv.hessian_cond_{min,lq,med,uq,max}`	minimum, lower quantile, median, upper quantile and maximum of the maximum divided by the minimum eigenvalue of the estimated hessian matrix

Table 2: Low-level features; summary of sub-features within the feature classes and assignment to feature groups used for predicting BBOB function groups and high-level function properties by means of the SMS-GA (cf. sections 4 and 5).

using “leave-one-function-out” resampling, which partitions the observations into 24 test sets corresponding to the BBOB functions.

Feature selection: Forward search adds features iteratively to the current set using a resampled estimate of a single performance measure, e.g. misclassification error. We initially considered this technique, but abandoned it very quickly, as features will only be chosen w.r.t. their predictive power, without regard for their computational cost. Instead, we opt to employ a multi-objective optimization approach, in which we try to minimize the misclassification rate while also minimizing the cost of calculating the active features and the total number of feature groups used. We use an adapted SMS-EMOA, as proposed by Beume et al. (2007), for our discrete parameter space. By replacing the simulated binary crossover and polynomial mutation operators with the well known uniform binary crossover and uniform mutation operators we obtain the resulting algorithm, which we called SMS-GA[2]. It generates not just one set of features, but many diverse ones with different trade-offs w.r.t. the above three criteria. The features are combined into feature groups (see Table 2), and each bit in the genome of the GA represents one such feature group. The rationale for this is that there are usually several ELA features which can be derived from one set of calculations. Therefore the optimization process can view them as one “feature”, since adding further features from the feature group will not increase the computational cost. This also reduces the size of the discrete search space.

Feature selection and all required algorithms of machine learning were implemented in R using mlr (Bischl, 2011).

4. FEATURE SELECTION FOR BBOB GROUPS

4.1 Experimental Setup (Problem 1)

A random forest (RF) is used to classify the $24 \times 5 \times 5 \times 5 = 3000$ function instances (# functions × # function instances × # dimensions × # repetitions) of the BBOB contest (cf. to Hansen et al. (2009) for details) into the five predefined BBOB groups based on the low-level features described in section 2.2. Features are selected with the purpose of extracting the most relevant ones for characterizing the fitness landscape of a given problem. In order to investigate the influence of the size s of the initial LH design, nine parallel computations of the low-level features have been conducted for all elements of s = {5000, 2500, 1250, 625, 500, 400, 300, 200, 100}. Each low-level feature group based on D^s is encoded by one bit which can be active or switched off, resulting in a bit string of length 90. Each group name is supplemented by the elements of s, the first nine elements of the bit string encode the features (`approx.linear_5000`, ..., `approx.linear_100`).

The three quality criteria for the SMS-GA optimization are specified as follows: (1) The random forest is cross-validated on the function instances as described in section 3, and the MCE is used to evaluate its predictive quality. (2) In case of extremely time-consuming fitness function evaluations, it is impossible to evaluate a huge number of sample points which would rule out the use of e.g. the local search features. Thus, the computational complexity of the low-level features is considered as well. It is reflected by the number of fitness function evaluations (NFE) required for the computation of the features. Details are provided in Table 3. (3) Model complexity is addressed by the number of selected features (NSF).

All three criteria are linearly transformed to the interval [0,1] by dividing each of them by its maximal value in order to ensure comparable scales and not to favor a criterion during the multi-objective optimization.

The SMS-GA is utilized for simultaneously minimizing the three criteria MCE, NFE and NSF operating on the 90 dimensional bit string as described above. This Pareto front approximation facilitates the selection of a solution with respect to available computational resources and time constraints.

Ten runs of the SMS-GA are executed in parallel using a population size of $\mu = 20$, 2000 fitness evaluations, nadir point $nd = (1, 1, 1)$, uniform binary crossover with probabil-

[2]An R implementation of the SMS-GA can be obtained from `http://ptr.p-value.net/gco10/sms_ga.R`

Feature Group	Cost of selected bits
convex	$\sum_{i=1}^{l} \tilde{s}_i + l \cdot 1000$
ls	$\sum_{i=1}^{l} (\tilde{s}_i + 50 \cdot d \cdot ls_{\bar{FE}}(s_i))$
numderiv.grad	$\sum_{i=1}^{l} \tilde{s}_i + l \cdot 100 \cdot d^2$
numderiv.hessian	$\sum_{i=1}^{l} \tilde{s}_i + l \cdot 100 \cdot d^3$
all others	$\sum_{i=1}^{l} \tilde{s}_i$

Table 3: Cost (i.e. NFE) of the selected bits. The vector $\tilde{s} = (\tilde{s}_1, \ldots, \tilde{s}_l)$ contains the different initial design sizes within the selected features; $ls_{\bar{FE}}$ reflects the mean number of FE of all local searches for the respective initial design; d equals the dimensionality of the test function.

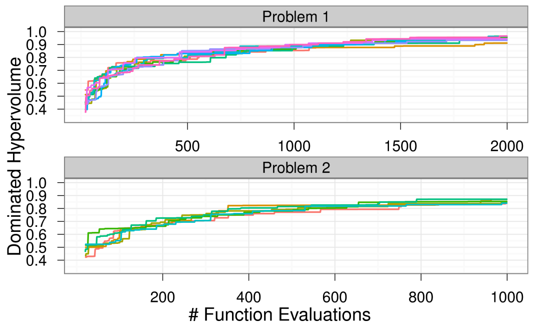

Figure 2: Plot of the hypervolume dominated by the active population after each function evaluation for the two optimization problems (10D). The colors denote the different runs of the SMS-GA.

ity $p_c = 0.25$ and a mutation rate of $p_m = 0.05$. The latter two parameters are set in accordance with usual recommendations in the GA literature. All members of the archives of non-dominated solutions are merged and the non-dominated solutions within this set of points are determined, which represent the final approximation of the Pareto front of the problem.

In Mersmann et al. (2010) it was shown that the performance of the competing algorithms in the BBOB contest differs widely for lower (2-3) and higher dimensions (5-20). In order to analyze the effect of increasing dimension on the selection of the low-level features, the analysis is performed separately for the decision space dimensions $d = (5, 10, 20)$, since these are of primary interest for practical applications.

4.2 Results

The combined non-dominated solutions of the 10 runs for the 10D case are shown in Fig. 4. Looking at the plot, we see that for the most part features based on small initial designs are chosen. This alone is an encouraging result because it implies that using just a few function evaluations we can characterize the 24 BBOB test functions to such an extent, that we can, in the extreme case, perfectly predict to which BBOB function group they belong. One surprising aspect is that for some individuals the chosen feature set seems redundant. For example, the first solution with a misclassification rate of 0 and a cost of $10^{-2.4}$ uses both the `convex.(...)_625` and then `convex.(...)_5000` features. We believe that, given more time, the SMS-GA would likely eliminate one of the two. From Fig. 2 it becomes obvious that the SMS-GA has not finally converged with respect to the obtained hypervolume. We plan to analyze this further. Another important thing to note is that there appears to be no feature which is required regardless of the chosen trade-off. Therefore the optimization process seems to have generated truly diverse solutions, not just in the objective space but also in the parameter space.

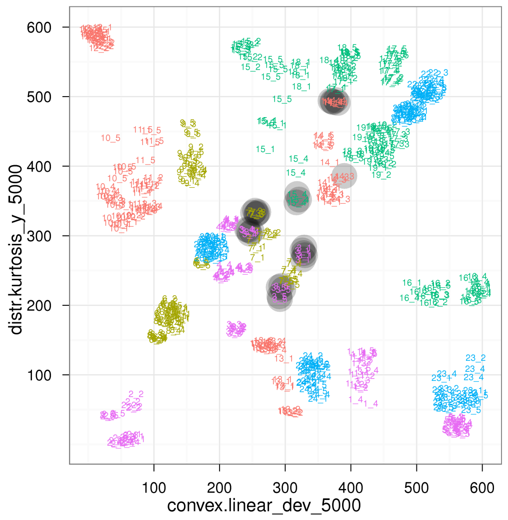

Figure 3: Scatter plot of two features, selected via forward selection and cross-validation on the function instances (10D), used to predict the BBOB function group. The color encodes the function group (labeled by function_instance) and the dark circles mark misclassified function instances during the cross-validation.

Fig. 5 visualizes the normalized ranks of important features within the BBOB groups, i.e. those features for which the feature groups are active in at least ten of the solutions generated by the SMS-GA. Though the rank structures are not clearly distinguishable in most cases, the highest differences can be detected for the "levelset" and "approx.linear" features, especially with respect to the fifth BBOB group. However, the effect of the problem dimension on the feature values is observable which justifies the separate analysis for the different dimensions.

Finally, let us point out that the chosen scenario might result in an optimistically biased misclassification rate. Given that we have differently parametrized instances of the same test function in both the training and test set, the job of extrapolating from the "known" behavior of the training set of test functions to the "unknown" test functions is much easier than if we had opted to leave out the whole set of test instances of a test function in each cross-validation fold. To illustrate this effect, we can consider a simplification of the optimization done in this section. We can use simple forward selection, as described in section 3, to find two relevant features to predict the BBOB function group of a test function. Fig. 3 shows such a pair of features for the 10D test functions. We clearly observe that the only misclassified instances, those with a dark circle behind them, are functions which are close to a cluster of other test function instances. But looking at the bottom left of the plot, we see an isolated cluster of test function instances all belonging to test function 2. If we had completely removed function 2 in the training phase, how could the classifier possibly know that functions in this region belong to the BBOB function group 1? A more likely scenario is that functions in this region of the plot would be assigned to classes 2 or 3. Results in the 5D and 20D case are similar and the same plots as shown here are provided for these two cases in the supplementary material.

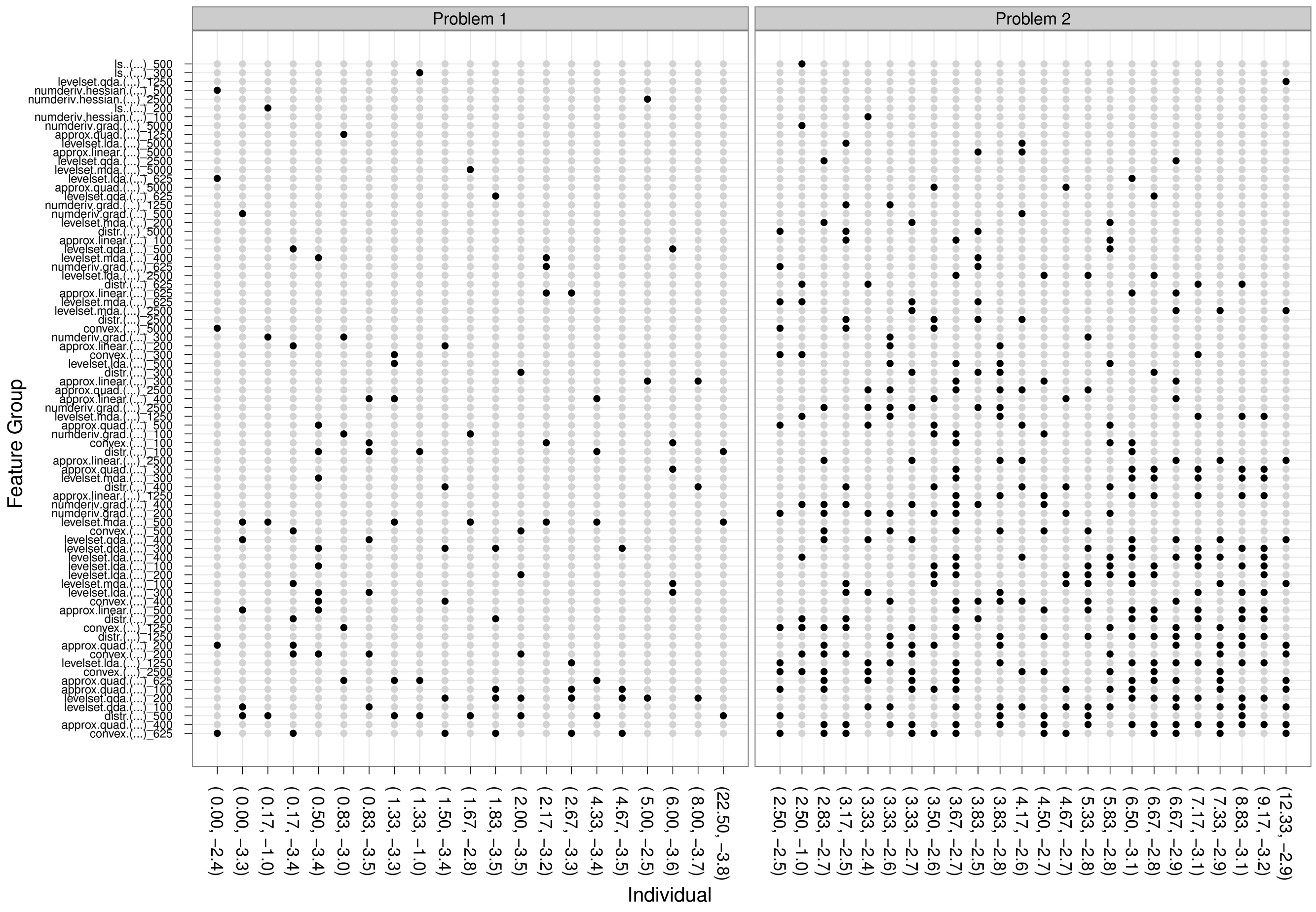

Figure 4: Bitplot of the overall best solutions (10D). Feature groups which are never selected are not shown to save space. The feature groups are ordered, from bottom to top, according to how frequently they occur in the solution sets. Each column represents one individual in the final set of non-dominated solutions. These solutions are again ordered according to the first two objectives. The axis label shows the MCE (in percent) and the cost (on $\log_{10}$ scale) of the solution. Black dots denote active features, grey dots mean the feature is not park of the solution vector.

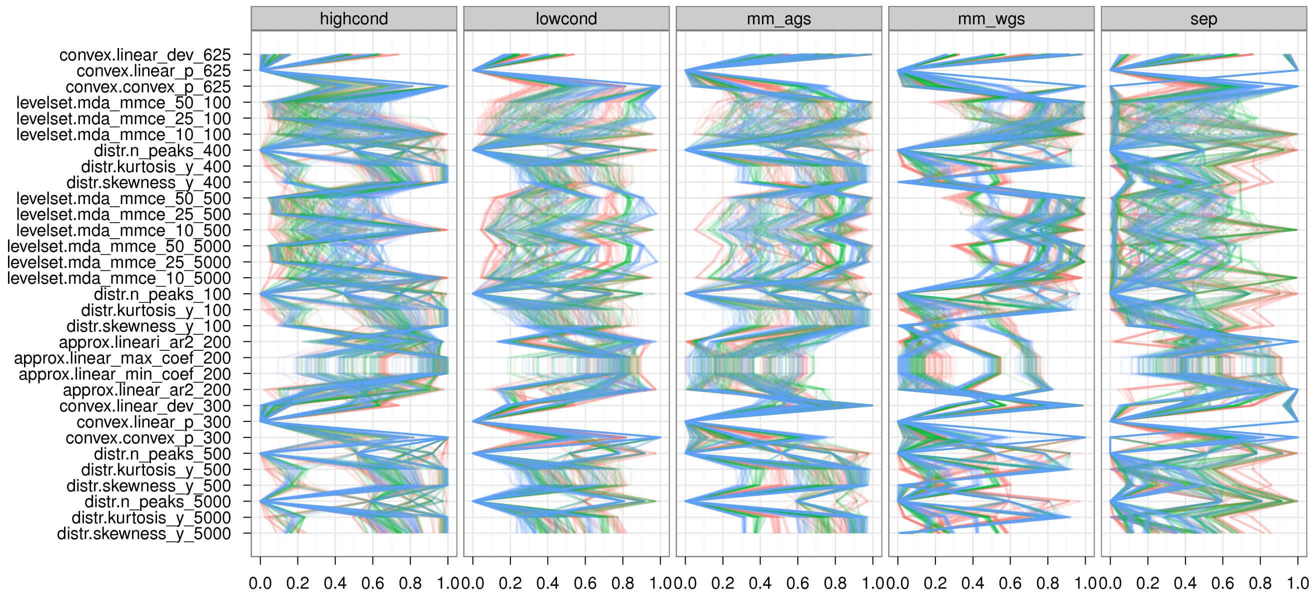

Figure 5: Parallel Coordinate Plot of the 31 features present in at least 10 of the final non-dominated solutions to the problem studied in section 4. Each line represents one calculation of the features on one BBOB test function instance (red: 5D, green: 10D, blue: 20D). The x-axis represents the ranks of the feature values within the groups scaled by the overall maximum rank.

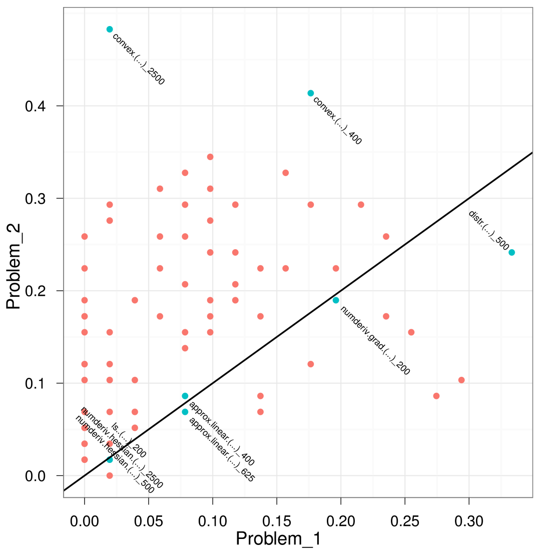

Figure 6: Probabilities of belonging to a final solution of problem 1 and 2 for all feature groups of Fig. 4.

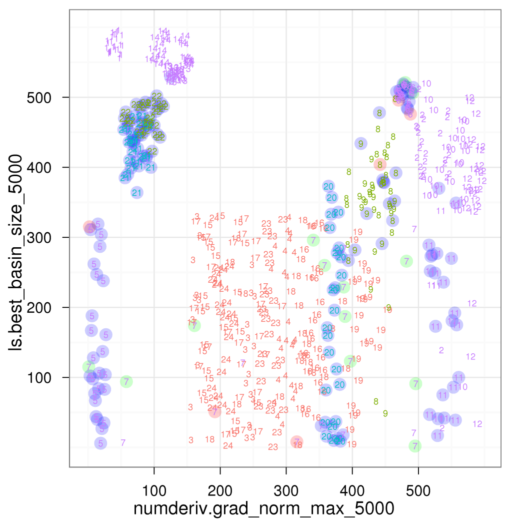

Figure 7: Scatter plot of the rank transform of two variables selected to predict the multi-modality property using leave-one-function-out. Each test function instance is represented by its BBOB test function number, its color encodes the multi-modality class from Tab. 1. The colored circles mark instances which are misclassified (red: incorrect in two class setting, blue: incorrect in four class setting, green: incorrect in both settings). Because the local search feature is equal to one for many of the functions, the first ≈ 350 ranks should be treated as equal. They are randomly assigned here to aid in the visualization.

5. FEATURE SELECTION FOR HIGH-LEVEL FEATURES

In general, knowledge of specific characteristics of the fitness landscape of a given problem is desired independent from the BBOB function grouping. More specifically, a direct prediction of function properties as presented in Table 1 such as e.g. the level of multi-modality has to be provided. Especially the ability to differentiate between multi-modal problems with and without global structure will facilitate the selection of an appropriate algorithm for an optimization problem at hand.

5.1 Experimental Setup (Problem 2)

The experimental setup is very similar to the one of section 4.1. The main difference is that in this case seven different classification problems are solved in parallel, i.e. the level of each high-level feature is predicted based on the low-level feature groups encoded in the 90-dimensional bit string.

The optimization criteria for the SMS-GA remain the same except for the computation of the misclassification error. The cross-validated MCE is computed for each of the seven classification problems, and the maximum of these errors (MaxMCE) is used as the quality criterion reflecting model accuracy in the multi-objective optimization problem. Therefore the optimization goal is to find sets of low-level features which offer optimal trade-offs for the simultaneous minimization of MaxMCE as well as NFE and NSF. Six runs of the SMS-GA with 1000 evaluations are conducted, and the analysis is again carried out separately for 5, 10 and 20 dimensions.

5.2 Results and Comparison of Feature Sets

The optimization task posed in this section is much harder since we are optimizing several classification problems simultaneously. This is reflected by the higher MaxMCE of the final non-dominated individuals. At the same time, by looking at Fig. 4, we see that the number of feature groups per individual is higher when compared to the problem in the previous section. Maybe even more surprising is the fact, that the feature groups used are not necessarily the same ones used in the other optimization problem. Since, given a function's high-level features, we can deduce the BBOB function group it belongs to, we would have expected the two solution sets to be more similar. This can also be seen from Fig. 6, in which the probabilities of belonging to a final solution for each feature group of Fig. 4 are visualized for both problems. Only few feature groups, located close to the main diagonal, have roughly equal probability of appearance in the solutions of both problems. The unbalanced number of active feature groups for both problems becomes obvious as well as much less points are located below the diagonal than above. One reason for the large number of active bits in the solutions of problem 2 might be that, again, the SMS-GA has not converged within the afforded number of function evaluations. Another possibility to further reduce the active bits of the solutions might be some kind of backward search at the end or even include such an operator in the GA. One should also keep in mind that possibly very different feature groups are helpful in predicting the seven properties.

Additionally, the feature groups with extreme probabilities are labeled for each problem. Whereas for problem 1 the features of the y - distribution group are of higher importance than the convexity features which is contrary to the situation in the second problem.

6. DISCUSSION AND OUTLOOK

A crucial aspect in optimization is the ability to identify the most suitable optimization algorithm for a (black-box) optimization problem at hand. This procedure can be based on key characteristics of the fitness landscape such as the degree of global structure or the number of local optima, both of which have proven to have a strong influence on the performance of different optimizers and allow to differentiate between them. As those properties are usually unknown for practical applications, we present an approach to automatically extract low-level features from systematically sampled points in the decision space. Examples of these are the probability of convexity, the estimated number of local optima based on local search techniques or the accuracy of fitted meta models.

The relevance of the proposed low-level features is proven by successfully predicting the predefined function groups of the BBOB'09/10 contest for all function instances based on the estimated low-level features with only marginal classification errors. Next to the accuracy of the predictions, the cost for calculating the features is considered as well as the model complexity measured by the number of chosen feature groups. Using multi-objective optimization techniques the Pareto front of this problem can be approximated, offering different trade-off solutions to choose from. In addition, and much more relevant for practical applications, a highly accurate prediction of high-level function features derived from expert knowledge is achieved independently of the BBOB function groupings. Contrary to our initial expectation, much fewer sample points are required to calculate the features needed for highly accurate classifications, making this approach, to a certain extent, applicable even for expensive real-world optimization problems.

In future work we will study how the performance of the optimization algorithms which took part in the BBOB contests relates to the proposed low-level features. The aim is to identify algorithms which perform well on different subsets of functions or function groups from the test set which in turn can be characterized by the computed low-level features. From there, a generalization to a new optimization problem should be possible by computing its low-level features and selecting the best performing algorithm for these characteristics in the BBOB competition.

As discussed in section 5, the scenario chosen for our optimization does not have to hold necessarily. It may in fact be desirable to extrapolate to completely new functions instead of just new instances of the same test function. We have done some preliminary work in this direction, using leave-one-function-out cross-validation. As an example of the results one can expect here, we again chose the two best variables to predict the multi-modality feature using a random forest with forward feature selection. Only the 50 features with $s = 5000$ are used, as we disregard their costs in this small experiment. We both consider the original 4-class problem and a 2-class version, where we try to predict whether the multi-modality property is "none" or not. The results for the 5D case are shown in Fig. 7. While we only achieve an accuracy of $\approx 73\%$ on the 4-class problem, we can offer some insights into why this is the case. Consider, for example, the function 5. It lies isolated in the bottom left corner of the plot. We conjecture that there are not enough "similar" functions in the BBOB test set for the classifier to possibly correctly assign this function to the "no multi-modality" class. On the other hand, the 2-class problem can be solved with an error of less than 4%. Surprisingly, using all 50 features in the random forest does not really improve the performance for the 4-class problem and even leads to degradation in the 2-class task.

While it is desirable for a test set used in a benchmarking setting to be as small as possible, it is a necessary requirement to have a few generally similar functions in the set from a machine learning perspective. We feel that finding such a "space filling" set of test functions is one of the great challenges of benchmarking. Using the low-level features proposed in this article as a tool to judge the distribution of the chosen test functions in the space of all possible test functions might be a viable way to approach this problem.

A promising perspective would therefore be the extension of the BBOB test set by functions with specific characteristics so that a more balanced distribution with respect to the levels of the high-level features is provided. The introduced low-level features could be used as the basis for a system that automatically generates test functions with certain desired properties, e.g. by means of genetic programming. Using such a system, methods from design of experiments could be used to systematically sample functions for predefined levels of the low-level features, resulting in a well balanced set of test functions. Lastly, the inclusion of real-world benchmarking scenarios is also highly desirable in order to minimize the gap between artificial test functions and the challenges of practical optimization tasks.

Acknowledgements.
This work was partly supported by the Collaborative Research Center SFB 823 (B4), the Graduate School of Energy Efficient Production and Logistics and the Research Training Group "Statistical Modelling" of the German Research Foundation.

References

BEUME, N., NAUJOKS, B., AND EMMERICH, M. 2007. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research 181*, 3 (September), 1653–1669.

BISCHL, B. 2011. mlr: Machine learning in R, http://mlr.r-forge.r-project.org.

BREIMAN, L. 2001. Random forests. *Machine Learning 45*, 1, 5–32.

HANSEN, N., AUGER, A., FINCK, S., AND ROS, R. 2009. Real-parameter black-box optimization benchmarking 2009: Experimental setup. Tech. Rep. RR-6828, INRIA.

HANSEN, N., FINCK, S., ROS, R., AND AUGER, A. 2009. Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA.

LINFIELD, G. R. AND PENNY, J. E. T. 1989. *Microcomputers in Numerical Analysis*. Halsted Press, New York.

MERSMANN, O., PREUSS, M., AND TRAUTMANN, H. 2010. Benchmarking evolutionary algorithms: Towards exploratory landscape analysis. In *PPSN XI: Proceedings of the 11th International Conference on Parallel Problem Solving from Nature*, R. Schaefer et al., Eds. Lecture Notes in Computer Science 6238. Springer, 71–80.

PREUSS, M. AND BARTZ-BEIELSTEIN, T. 2011. Experimental analysis of optimization algorithms: Tuning and beyond. In *Theory and Principled Methods for Designing Metaheuristics*, Y. Borenstein and A. Moraglio, Eds. Springer.

SUGANTHAN, P. N., HANSEN, N., LIANG, J. J., DEB, K., CHEN, Y.-P., AUGER, A., AND TIWARI, S. May 2005. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Tech. rep., Nanyang Technological University, Singapore. http://www.ntu.edu.sg/home/EPNSugan.