Research Article

Taleh Agasiev*

Characteristic feature analysis of continuous optimization problems based on Variability Map of objective function for optimization algorithm configuration

<https://doi.org/10.1515/comp-2020-0114>

Received Jul 06, 2019; accepted Feb 02, 2020

Abstract: Advanced optimization algorithms with a variety of configurable parameters become increasingly difficult to apply effectively to solving optimization problems. Appropriate algorithm configuration becomes highly relevant, still remaining a computationally expensive operation. Development of machine learning methods allows to model and predict the efficiency of different solving strategies and algorithm configurations depending on properties of optimization problem to be solved. The paper suggests the Dependency Decomposition approach to reduce computational complexity of modeling the efficiency of optimization algorithm, also considering the amount of computational resources available for optimization problem solving. The approach requires development of explicit Exploratory Landscape Analysis methods to assess a variety of significant characteristic features of optimization problems. The results of feature assessment depend on the number of sample points analyzed and their location in the design space, on top of that some of methods require additional evaluations of objective function. The paper proposes new landscape analysis methods based on given points without the need of any additional objective function evaluations. An algorithm of building a so-called Full Variability Map is suggested based on informativeness criteria formulated for groups of sample points. The paper suggests Generalized Information Content method for analysis of Full Variability Map which allows to get accurate and stable estimations of objective function features. The Sectorization method of Variability Map analysis is proposed to assess characteristic features reflecting such properties of objective function that are critical for optimization algorithm efficiency. The resulting features

are invariant to the scale of objective function gradients which positively affects the generalizing ability of problems classification algorithm. The procedure of the comparative study of effectiveness of landscape analysis algorithms is introduced. The results of computational experiments indicate reliability of applying the suggested landscape analysis methods to optimization problem characterization and classification.

Keywords: global optimization, continuous optimization problem, meta-optimization, algorithm configuration, parameter tuning, dependency decomposition approach, characteristic features, exploratory landscape analysis, variability map, generalized information content, sectorization method

1 Introduction

Modern systems of automated design and engineering analysis include programs that implement advanced algorithms of continuous global optimization. Due to the increasing complexity of engineering design, high efficiency of optimization algorithms becomes more significant. Most advanced global optimization algorithms have configurable parameters drastically affecting the algorithm behavior and efficiency of solving optimization problems. It is necessary to properly manage the configurable parameters of algorithms to achieve the best possible solution with the least amount of computation [1, 2].

Let us consider the following global continuous optimization problem q :

$$\min_{X \in D_X \subset \mathbb{R}^{|X|}} f(X) = f(X^*) = f^*, \quad (1)$$

where X is $|X|$ -dimensional vector of design variables; D_X is the domain of allowable X values; $f(X)$ is the objective function; X^* , f^* are the optimal design vector found and the corresponding objective value.

*Corresponding Author: Taleh Agasiev: Bauman Moscow State Technical University, Moscow, Russia; Email: agtaleb@mail.ru

Since it is critical to utilize computational resources effectively, methods of preliminary problem analysis are widely used before solving real-world and computationally expensive optimization problems [3]. One of the important results of such analysis is vector $C = (c_1, \dots, c_{|C|}) \in D_C$ of problem's q characteristic features, where D_C is a domain of possible values of features defined by *characterization algorithm* used. Besides the features that follow directly from q problem definition (1), e.g. the search space dimensionality $|X|$, vector C may also include features of objective function $f(X)$ [4], e.g. some ruggedness and multimodality metrics. The vector of features can include continuous, integer or categorical features. For simplicity's sake, we only consider continuous features, so that $D_C \subset \mathbb{R}^{|C|}$.

Characteristic features are assessed according to characterization algorithm by an expert and/or automatically, based on *landscape sample*, which is the result of $f(X)$ function evaluations at some points of the search domain D_X . Thus resulting values of features $c_1, \dots, c_{|C|}$ depend on the number and the location of points of the landscape sample which represents the discretized landscape of the function $f(X)$. The points for evaluation are usually chosen in accordance with a certain design of experiment method [5]. In the case of high computational complexity of objective function, it is reasonable to apply adaptive design of experiment methods [6]. The landscape sample points after being processed for the assessment of problem features can be used at the initialization stage of optimization algorithm, for example, to build an initial approximating model of an objective function in the case of Surrogate Based Optimization [7].

Let us denote $B = (b_1, \dots, b_{|B|})$ vector of configurable parameters of optimization algorithm and call it algorithm's *strategy*. The strategy can include simple continuous, integer or categorical parameters as well as complex control parameters, for example, the rule of adaptive changing of simple parameters values during the problem solving process [8]. Let us assume for the sake of clarity that all the parameters are continuous and $D_B \subset \mathbb{R}^{|B|}$. The methods presented in this work can also be applied when the vector B is mixed, i.e. includes parameters of different types.

The article is partially based on and extends the conference paper [9]. In addition to that, it presents a new approach to optimization algorithm configuration with characteristic features assessment powered by Variability Map based methods of exploratory landscape analysis. In the last part of this work, the procedure of assessing efficiency of landscape analysis algorithms is proposed and the computational experiment results are presented.

2 Optimization algorithms configuration

To configure optimization algorithms successfully one should define the proper way of measuring the efficiency of solving optimization problems. There are various types of efficiency metrics being used to compare solution quality as well as convergence rate of different algorithms or different strategies of a single algorithm [10]. Strictly speaking, the algorithm efficiency strongly depends on characteristic features C of the problem q being solved and on the vector of chosen algorithm strategy B .

In the case of solving computationally expensive real-world problems, the algorithm's user has to consider and formalize in some way the amount of computational resources available to find a certain solution. Basically, computational performance can be formalized in the maximum available number I of objective function evaluations. Reaching the maximum evaluations number I is the most common termination criteria for optimization algorithms designed for computationally expensive problems. Thus, solution quality, e.g. the best reached objective value within the given number of evaluations, is the most appropriate way of measuring and comparing the effectiveness of such algorithms.

Let us define efficiency indicator $e(B, C, I)$ measuring the quality of solution $f(X^*)$ obtained by solving problem $q(C)$ with a given strategy B in a predefined number of evaluations I . The problem of finding the best algorithm strategy B^* for solving an optimization problem $q(C)$ in a given I number of objective function evaluations is called *meta-optimization problem*:

$$\min_{B \in D_B \subset \mathbb{R}^{|B|}} e(B, C, I) = e(B^*, C, I). \quad (2)$$

It is reasonable to consider various efficiency criteria formalized in vector efficiency indicator $E = (e_1, e_2, \dots)$, e.g. it can be the mean and variance of objective values f^* found by random starts of an algorithm with a certain strategy B . In that case, finding the best strategy of an algorithm is a multi-objective optimization problem itself. Let us for the sake of simplicity consider a scalar efficiency indicator e , corresponding to the average of the best reached values f^* . All the methods introduced here can be generalized for vector efficiency indicator E .

The overall goal of an algorithm configuration is to be able to determine B^* for any combination of C and I values, i.e. to build a map $(C, I) \mapsto B^*$. Obviously, solving a meta-optimization problem (2) for certain C and I values by applying an optimization algorithm (which becomes a

meta-optimization algorithm in this context) is only appropriate for computationally cheap optimization problems $q(C)$. On the other hand, meta-optimization as such is not relevant if a certain problem $q(C)$ can be solved reasonably fast and repeatedly with different algorithm strategies B . Meta-optimization problem (2) of finding the best strategy B^* can not be solved "directly" for real-world optimization problems (1).

The ultimate way of solving most kinds of algorithm configuration problems is to build an approximation model $\hat{e}(B, C, I)$ predicting algorithm efficiency. To do so, it is necessary to make a strong assumption that there is a smooth dependency to be revealed, that is algorithm efficiency $e(B, C, I)$ does not vary considerably for very close C and B values (which is usually the case for close I values). The assumption becomes even stronger in the case of building $\hat{B}^*(C, I)$ approximation to predict efficient strategies B^* directly without considering optimization algorithm's efficiency e . However, any of these can not be guaranteed since the nature of $e(B, C, I)$ dependency and its complexity is different for various parameters B to be tuned and for features C chosen for problem characterization.

Building $\hat{e}(B, C, I)$ approximation directly is a challenging and controversial decision. Even with the use of powerful state-of-the-art machine learning methods, it needs a significant number of experiments performed in B and C spaces for various I values. Another problem is to iteratively refine such a model in the case of obtaining additional information on efficiency of the algorithm being configured, e.g. if the algorithm efficiency $e(B, C, I)$ is assessed for other B , C or I values during its exploitation.

There are different possible ways to reduce computational complexity of building $(C, I) \mapsto B^*$ map at the expense of accuracy of resulting predictions:

- discretize D_B space so that the algorithm configuration problem is narrowed down to the strategy or "algorithm" selection problem [11],
- discretize D_C space and find B^* value for groups or, strictly speaking, classes of optimization problems separately [12],
- fix the number of evaluations I , which is rarely even considered in literature (e.g. it is considered in [13]).

A comprehensive overview of existing approaches to automated algorithm selection and configuration was presented in [14]. All these approaches, as well as building $e(B, C, I)$ function approximation directly, have some obvious drawbacks preventing the accurate prediction of B^* strategies, on one side, or requiring an inordinate amount of computations, on the other.

In this paper, another indirect approach of $e(B, C, I)$ function approximation and B^* strategy prediction is proposed to address the above restrictions and establish practically applicable solution to configure optimization algorithms for solving real-world optimization problems.

2.1 Dependency Decomposition approach for algorithms configuration

Let us distinguish main stages of the general workflow of algorithm configuration based on approximation models.

Exploration stage is required for preliminary analysis of optimization algorithm's behaviour with various strategies $B \in D_B$ based on results of computational experiments performed with computationally cheap test optimization problems [15]. The results are used to build approximation models either predicting efficiency of the algorithm being analyzed or predicting the best strategies B^* in some way, e.g. for each class of test problems used.

Exploitation stage is the stage at which all the gathered information and models built are used to predict the best strategy B^* for solving a certain optimization problem q' in the following way:

1. perform a number of $f(X)$ function evaluations utilizing a portion of maximum evaluations number I' to obtain a landscape sample,
2. based on landscape sample points, calculate the problem features C' by the characterization algorithm chosen,
3. given the vector C' find or predict the strategy B^* for solving the problem q' within I' number of evaluations (including the number of landscape sample points).

Note that the exploration stage is not necessarily to be finished before the exploitation stage, it can be run permanently to gradually increase the accuracy of predictions over time.

Most of traditional approaches assume building a whole model $\hat{e}(B, C, I)$ or its components at exploration stage and predicting the algorithm efficiency to determine B^* on the exploitation stage. The idea of Dependency Decomposition approach is to re-balance computational efforts of building $\hat{e}(B, C, I)$ approximation while saving it smooth for meta-optimization purposes and continuous in both B and C spaces. We suggest Dependency Decomposition method with the following steps:

1. At exploration stage:

- 1.1. Prepare a set of test problems $\{q_t\}$ based on expert knowledge of the target real-world problems.
- 1.2. Choose a characterisation algorithm and calculate characteristic features $\{C_t\}$ for all the test problems.
- 1.3. Collect a set of strategies $\{B_j\} \subset D_B$ for testing: use a design of experiment method to evenly cover the D_B domain, also include promising strategies $\{B_{t,k}^*\}$ found by solving meta-optimization problems (2) for each test problem $\{q_t\}$ with given $\{I_k\} \subset D_I$ values.
- 1.4. Solve all the test problems $\{q_t\}$ by each B_j strategy with given $\{I_k\}$ values, calculate relative efficiency indicator $e_{t,k}^r$ of the B_j strategy as

$$e_{t,k}^r = \frac{e(B_j, C_t, I_k) - e(B_{t,k}^*, C_t, I_k)}{\max(|e(B_{t,k}^*, C_t, I_k)|, 1)}, \quad (3)$$

where $e(B_j, C_t, I_k)$ indicates the best reached objective value f^* when using the B_j strategy for solving q_t problem in I_k evaluations.

- 1.5. For each B_j strategy based on $\{C_t, I_k, e_{t,k}^r\}$ sample build an approximation model $\hat{e}^r(B_j, C, I)$, $C \in D_C$, $I \in D_I$ predicting its efficiency for any C and I values.

2. At exploitation stage:

- 2.1. For a new given problem q' calculate the vector C' of its features according to the characterization algorithm chosen at step 1.2.
- 2.2. Using $\hat{e}^r(B_j, C, I)$ models built at step 1.5 predict efficiency $\hat{e}_j^r(B_j, C', I')$ of each considered strategy B_j given the C' vector and the specified I' value.
- 2.3. Based on $\{B_j, \hat{e}_j^r\}$ sample build $\hat{e}^r(B, C', I')$, $B \in D_B$ model to predict the algorithm efficiency in continuous B space for given C' vector and I' value.
- 2.4. Find the best predicted strategy $\hat{B}^*$ by solving meta-optimization problem (2)

$$\min_{B \in D_B \subset \mathbb{R}^{|B|}} \hat{e}^r(B, C', I') = \hat{e}^r(B^*, C', I'). \quad (4)$$

The scheme of Dependency Decomposition method with its main steps is shown in Figure 1.

One of the key issues to be solved is proper comparison of the quality of solutions for various problems q_t having a different scale of objective function values $f(X)$. The comparison may be done in relative units if the solutions are known which is not always the case. For this reason the

Figure 1: The main steps of Dependency Decomposition method

best objective value f^* reached by the best found strategy B^* is used as the reference point in the metric (3).

The general scheme of predicting the efficiency of a certain strategy is illustrated in Figure 2.

Figure 2: Predicting efficiency of a certain strategy B' for solving a problem with C' vector of features in a given I' number of evaluations (example for $|B| = 1$ and $j = 1, 2, 3$)

Each model $\hat{e}_j^r(B_j, C, I)$, $C \in D_C$, $I \in D_I$ is a slice of the whole $e(B, C, I)$ function across B axes. The example of 3 slicing models and the $\hat{e}^r(B, C', I')$, $B \in D_B$ model built on exploitation stage with one-dimensional C and B vectors for a certain I value is shown in Figure 3.

Let us consider pros and cons of the method proposed. It has the following advantages:

- it provides a way to enhance incrementally the quality of predictions by adding more slicing models;
- it allows to build and refine slicing models independently without affecting the other ones;

Figure 3: Plots of slicing models built for B_1, B_2, B_3 strategies and $\hat{e}^r(B, C', I')$, $B \in D_B$ model predicting efficiency of solving $q(C')$ problem (example for $|B| = 1$, $|C| = 1$ and fixed I' value).

- it makes possible the usage of different approximation algorithms at different stages;
- it reduces computational efforts by making final predictions only for certain C' and I' values using a "locally" built model;

There are obvious disadvantages of such approach:

- additional computational efforts are required for building the final model at the exploitation stage;
- the accuracy of predictions is sensitive to the position of slicing models;
- it requires managing prediction accuracy of slicing models separately.

Generally speaking, the method reduces the complexity of approximating the $e(B, C, I)$ dependency by decomposing it between the stages. It allows to control the quality and complexity of the approximation models along C and B dimensions separately, which is critical given that the algorithm efficiency varies in a different manner along these dimensions. For example, one can use approximation algorithms that better handle noised dependencies (e.g. Random Forest) at the exploration stage and build a smooth model which is suitable for solving meta-optimization problem (e.g. Gaussian process model) at the exploitation stage. The accuracy of slicing models strongly depends on the characterization algorithm used to assess optimization problem features.

One can obviously slice across I axis as well building $\hat{e}^r(B_j, C, I_k)$, $C \in D_C$ models at the exploration stage and predicting the efficiency dependent on both B and I values at the exploitation stage (Figure 4). That will in fact reduce the complexity of slicing models and increase their total number as well as the complexity of the final model $\hat{e}^r(B, C', I)$, $B \in D_B$, $I \in D_I$ built on the fly for a certain C'

vector at the exploitation stage. That way of dependency decomposition may be suitable if the maximum number of evaluations I is also under consideration and to be determined eventually. Most often it is the pre-defined parameter and there is no special need of such slices since the meta-optimization is only performed in B space.

Figure 4: Dependency Decomposition method scheme in case of additional slicing across I axis (example for 9 slicing models: for 3 vectors B and 3 values I)

Another possibility is to make slices along C axes at the exploration stage and build models $\hat{e}^r(B, C_t, I)$, $B \in D_B$, $I \in D_I$ predicting the efficiency of strategies when solving each given q_t problem respectively. This approach seems to be more natural with regard to meta-optimization procedure performed for each test problem q_t at step 1.3 of the exploration stage and there is no need in using relative efficiency indicator (3) as well. One can use the final $\hat{e}^r(B', C, I')$, $C \in D_C$ model, for example, to explore the efficiency of a certain B' strategy for solving different classes of problems q . Such a way of dependency decomposition is still not suitable for finding the best strategy B^* of solving real-world optimization problems. There is also an option of building $\hat{e}^r(B, C', I')$, $B \in D_B$ model for a certain q' problem using the predictions of $\hat{e}^r(B, C_t, I)$ models weighted in some way by the distance from C' vector to $\{C_t\}$ vectors (Figure 5). To make this feasible, the proper design of characteristic features' vector is required, so that the efficiency of different strategies vary similarly when solving problems that correspond to vectors close in the space of features.

The critical part of Dependency Decomposition method, as it is for any algorithm configuration method, is the characterization algorithm being used for illuminating significant features of optimization problems. Ensuring acceptable generalization capability of $\hat{e}^r(B_j, C, I)$ models requires development of effective characterization algorithm to preserve the dependency smooth along C dimensions.

Figure 5: Plots of slicing models built for c_1, c_2 values and $\hat{e}^r(B, C', I')$, $B \in D_B$ model predicting efficiency of solving $q(C')$ problem (example for $|B| = 1, |C| = 1$ and fixed I' value).

3 Characteristic feature analysis of optimization problem

Methods of characteristic feature analysis of optimization problem and its objective function in particular are called Exploratory Landscape Analysis (ELA) methods [16]. Some of these methods require performing a certain number of objective function evaluations based on certain inner rules (e.g. random or adaptive walk [17]). That approach is not applicable for problems with high computational complexity of objective function evaluations. Hence, we only consider ELA methods that assess objective function features without making any additional evaluations [18–20]. The outcome of these methods is only based on the given results of objective function evaluations, i.e. on the landscape sample containing a certain number of design points $X_i \in D_X$ and corresponding objective values $f_i, i \in [1 : n]$ calculated preliminarily. Strictly speaking, that kind of ELA methods assess the landscape sample features above all, on the basis of which assumptions on objective function properties are made. Reliability of the assumptions depends on the number n of landscape sample points and their location in the D_X domain. A proper landscape sample is supposed to evenly cover the D_X domain taking into account $f(X)$ function properties as well. Therefore, it is more appropriate to use advanced adaptive design of experiment methods to get reliable estimations of the objective function features. That is ELA methods can also be distinguished according to the sensitivity of assessments to minor changes in the location of landscape sample points. Information Content of Fitness Sequences (ICoFiS) method [21] is, in our view, one of the methods that is more stable to landscape sample variations. The Variability Map

of objective function and some of landscape analysis methods presented in this section were inspired by the concept of ICoFiS method that allows to assess ruggedness and multimodality of objective function.

3.1 Information Content of Fitness Sequences

Let us briefly describe the main steps of ICoFiS method that assess features of the objective function $f(X)$ based on the landscape sample points $X_i, i \in [1 : n]$:

1. Build a path connecting all the points in a certain order (e.g. by moving randomly or by choosing the closest uncovered point as the next)
2. Calculate differences δ_i between objective values corresponding to neighbouring points along the path built

$$\delta_i = \frac{f_{i+1} - f_i}{\|X_{i+1} - X_i\|}, i \in [1 : n - 1] \quad (5)$$

3. Convert the sequence of δ_i values into a symbol sequence $S(\epsilon) = \{s_1(\epsilon), \dots, s_{n-1}(\epsilon)\}$ given that

$$s_i(\epsilon) = \begin{cases} \bar{1}, & \delta_i < -\epsilon, \\ 0, & |\delta_i| \leq \epsilon, \\ 1, & \delta_i > \epsilon, \end{cases} \quad (6)$$

where ϵ is a parameter setting sensitivity of the method to the scale of objective values changes.

4. Calculate the features that measure the ruggedness of the objective function based on values of *information content* function

$$ic(\epsilon) = - \sum_{a \neq b} p_{ab}(\epsilon) \log_6 p_{ab}(\epsilon), \quad (7)$$

$$p_{ab}(\epsilon) = \frac{n_{ab}(\epsilon)}{n - 2}, \quad (8)$$

where $a, b \in \{\bar{1}, 0, 1\}$ compose the *block* of ab type of consecutive s_i, s_{i+1} elements (e.g. $\bar{1}\bar{1}, \bar{1}0, 01, 11$ blocks in $S(\epsilon) = \{1, \bar{1}, 0, 1, 1\}$); p_{ab} is the assessment of probability of ab type blocks detection in $S(\epsilon)$; n_{ab} is the number of ab type blocks in $S(\epsilon)$.

5. Calculate the features that measure the smoothness of the objective function based on values of *partial information content* function

$$icp(\epsilon) = \frac{|S'(\epsilon)|}{n - 2}, \quad (9)$$

where $S'(\epsilon)$ sequence is formed by removing all the 0 elements and repetitions from $S(\epsilon)$ (e.g. $S'(\epsilon) = \{1, \bar{1}, 1\}$ for $S(\epsilon) = \{1, \bar{1}, 0, 1, 1\}$).

Essentially, ICoFiS method characterizes features of one-dimensional piece-wise linear function produced by ordering the points of landscape sample. Thus, the accuracy of estimations depends above all on the path-building algorithm applied.

Coordinates of some notable points of $ic(\epsilon)$ and $icp(\epsilon)$ functions are used as components of vector C of problem characteristic features. Examples of such points for Rosenbrock and Ackley test functions [15] are shown in Figure 6. Some of the rules defining notable points and their interpretation were proposed in [21].

Figure 6: Examples of notable points of $ic(\epsilon)$ and $icp(\epsilon)$ functions calculated for landscape samples of Rosenbrock and Ackley test functions [9].

Low computational complexity of test optimization problems allows to explore the objective landscape extensively using the results of a large number of function evaluations. However, in most practically useful cases, the landscape sample is small and probably irregular. This raises the issue of sample ordering points optimal to get the most representative mapping of function's landscape to one-dimensional space and subsequently achieve better accuracy of feature assessments.

3.2 Variability Map of objective function

Let us introduce an alternative formulation of ICoFiS method with a view to apply it without preliminary ordering of landscape sample points [9]. The original method analyses blocks of certain types consisting of 2 consecutive elements of $S(\epsilon)$ sequence where each block in turn corresponds to 3 neighbouring (along the path) points of the landscape sample. Let us denote as $T^j, j \in [1 : l]$ blocks formed by ordered triples of design points $X_{i_1}^j, X_{i_2}^j, X_{i_3}^j$, where l is the total number of blocks, $i_1, i_2, i_3 \in [1 : n]$,

$i_1 \neq i_2 \neq i_3$. Each block T^j also refers to δ_1^j, δ_2^j and s_1^j, s_2^j values (for a certain ϵ value):

$$s_1^j(\epsilon) = \begin{cases} \bar{1}, & \delta_1^j < -\epsilon, \\ 0, & |\delta_1^j| \leq \epsilon, \\ 1, & \delta_1^j > \epsilon, \end{cases} \quad s_2^j(\epsilon) = \begin{cases} \bar{1}, & \delta_2^j < -\epsilon, \\ 0, & |\delta_2^j| \leq \epsilon, \\ 1, & \delta_2^j > \epsilon, \end{cases} \quad (10)$$

$$\delta_1^j = \delta_1(T^j) = \frac{f(X_{i_2}^j) - f(X_{i_1}^j)}{\|X_{i_2}^j - X_{i_1}^j\|}, \quad (11)$$

$$\delta_2^j = \delta_2(T^j) = \frac{f(X_{i_3}^j) - f(X_{i_2}^j)}{\|X_{i_3}^j - X_{i_2}^j\|}.$$

In the original method, l equals $n - 2$ for ordered point $X_i, i \in [1 : n]$ of the landscape sample. Each block T^j corresponds to subsequent points along the path built. In accordance with Eq. 7 the results of $ic(\epsilon)$ function evaluations are not dependent on T^j blocks ordering but on $X_{i_1}^j, X_{i_2}^j, X_{i_3}^j$ points ordering within each block T^j . Thus, blocks can be formed independently without landscape sample ordering, which means that any triple of points of unordered landscape sample could be considered as a block. The "quality" of blocks formed that way and their total number l have a major impact on the correctness of characteristic feature estimations. Blocks could obviously include more design points with consequent changes in the original methodology.

As the set of blocks $\mathbb{T} = \{T^j, j \in [1 : l]\}$ is formed, the set $\Delta(\mathbb{T}) = \{(\delta_1^j, \delta_2^j); j \in [1 : l]\}$ composes a Variability Map (VM) of objective function's landscape. A good visual way of presenting the VM is to plot the points in δ_1, δ_2 coordinates [9]. Edges of the VM can be set by maximum and minimum δ_1^j, δ_2^j values.

The ϵ value of ICoFiS method defines the partition of the VM into 9 domains corresponding to different types of blocks T^j (Figure 7). For example, for all the T^j blocks represented by δ_1^j, δ_2^j points inside the central square area, both s_1^j, s_2^j symbols are of 0 value. Thus it can be said that the values of $ic(\epsilon)$ function characterize information content of the given VM partition. The values of $icp(\epsilon)$ function in contrast strongly depend on the landscape sample ordering and can not be considered within such a framework.

The accuracy of feature estimations based on VM is largely dependent on the block building method defining the rules of design points grouping. One example of such a method is preliminary ordering of landscape sample points as suggested by the authors of ICoFiS method.

Figure 7: Variability Map of objective function divided into 9 separate areas, each corresponding to a certain type of T^j blocks defined by s_1^j, s_2^j elements for a given ϵ value [9].

3.3 Full Variability Map of objective function

To be able to choose the most appropriate triples of points without ordering, let us introduce the block's quality criteria based on distances between design points and angles assumed by design points.

Distance criteria

It is reasonable to limit the maximum distance between points composing a T^j block [9]. Let d_{i_1, i_2}^j be the euclidean distance between $X_{i_1}^j$ and $X_{i_2}^j$ points, and d_{i_2, i_3}^j - between $X_{i_2}^j$ and $X_{i_3}^j$ points. Let us denote d_s the maximum allowable distance between points so that $d_{i_1, i_2}^j < d_s$ and $d_{i_2, i_3}^j < d_s, j \in [1 : l]$.

According to the distance criteria, increasing the quality of blocks requires minimizing d_s value so that each block consists of points locally grouped in the design space:

$$\min_{i_1, i_2, i_3 \in [1 : n]} d_s(X_{i_1}^j, X_{i_2}^j, X_{i_3}^j), j \in [1 : l] \quad (12)$$

Angle criteria

It is clear that the reliability of the information provided by T^j block on function's landscape correlates with the angle α^j assumed by $X_{i_1}^j, X_{i_2}^j, X_{i_3}^j$ points. For example, any block T^j is completely uninformative if $X_{i_3}^j$ closely approaches $X_{i_1}^j$ and α^j tends to 0. Conversely, blocks with correspond-

ing α^j values close to π appeared to be more preferable for consideration.

According to the angle criteria, increasing the quality of blocks requires minimizing cosine distance between $X_{i_2}^j X_{i_1}^j$ and $X_{i_2}^j X_{i_3}^j$ vectors so that α^j tends to π :

$$\min_{i_1, i_2, i_3 \in [1 : n]} \frac{\langle X_{i_2}^j X_{i_1}^j, X_{i_2}^j X_{i_3}^j \rangle}{\|X_{i_2}^j X_{i_1}^j\| \cdot \|X_{i_2}^j X_{i_3}^j\|}, j \in [1 : l] \quad (13)$$

Block building algorithm

Taking into account the quality criteria of blocks we can build a Full Variability Map (FVM) composing most informative blocks from the landscape sample points in the following way:

1. calculate the maximum allowable distance d_s between the points of blocks

$$d_s = \max_{i_1} \min_{i_2} d_{i_1, i_2}, i_1, i_2 \in [1 : n], i_1 \neq i_2; \quad (14)$$

2. for each point $X_i, i \in [1 : n]$ of the landscape sample;

- (a) find the points $X_{p_k}, k \in [1 : K]$ meeting the condition

$$d_{i, p_k} < d_s; \quad (15)$$

- (b) if $K > 1$ make new blocks $T^k, k \in [1 : K - 1]$ up of $X_{p_k}, X_i, X_{p_{k+1}}$ or $X_{p_{k+1}}, X_i, X_{p_k}$ points according to the chosen ordering.

- (c) calculate α^k value for each new T^k block and only keep the blocks with corresponding $\alpha^k \geq \pi/2$.

3. for all the kept blocks $T^j, j \in [1 : l]$ calculate δ_1^j, δ_2^j values by formula (11);

The ordering of points forming a block T^j defines signs of δ_1^j, δ_2^j values and hence the location of the corresponding point on the VM. Possible kinds of blocks with regard to the angle formed by $(X_{i_1}^j, f_{i_1}^j), (X_{i_2}^j, f_{i_2}^j), (X_{i_3}^j, f_{i_3}^j)$ points and their expected locations on VM are shown in Figure 8 for one-dimensional function $f(X)$.

Considering the two ordering directions, each triple of points forms two blocks which are symmetrically disposed on the VM in relation to $\delta_1^j = -\delta_2^j$ line: above the symmetry line ($\delta_1^j + \delta_2^j > 0$); below the symmetry line ($\delta_1^j + \delta_2^j < 0$). To examine the whole variety of blocks and landscape transitions, both of the ordering directions should be inspected for each triple of design points.

Variability Maps can also be a basis for developing metrics of landscape similarity of different functions. For example, special metrics are required to assess the quality

Figure 8: Variability Map (a) with symmetric points referring to different ordering of triples for each kind of blocks (b) [9].

of approximation models if there is no validation sample, which is usually the case for computationally expensive functions. In the case of surrogate based optimization, it is also crucial to select the objective function modelling technique properly [22]. The results of using FVM for advanced landscape validation of surrogate models and modelling technique selection will be presented in future works.

4 Methods of FVM analysis

Any FVM analysis method defines a characterizing function, values of which, as well as the values of its arguments, can be used as components of the vector C of problem's characteristic features. Characterization algorithms are produced on the basis of FVM analysis method by setting values of the corresponding characterizing function that will be used as components of C vector. For example, $ic(\epsilon)$ function is the characterizing function of ICoFiS method which is considered as the method of Variability Map analysis in section 3.2.

The general template of methods of assessing optimization problem features based on the results of FVM analysis has the following steps:

1. collect the landscape sample points $X_i, i \in [1 : n]$ and calculate the corresponding objective values f_i ,
2. build a FVM for the landscape sample using the algorithm provided in section 3.3,
3. according to the chosen characterization algorithm calculate the required values of the characterizing function (or its arguments),
4. compose the resulting vector C of problem's features based of the values obtained.

In sections 4.1 and 4.2 we introduce new FVM analysis methods which are consistent with the template described above. Characterization algorithms based on each of them will be presented in section 5.1.

4.1 Generalized Information Content method

Generalized Information Content method [9] has the same steps as the template method of assessing optimization problem features. Taking into account the symmetry of FVM, let us introduce ϵ_1 and ϵ_2 parameters to have more control of its partitioning into domains (Figure 9) compared to the original method with a single ϵ parameter.

Figure 9: Variability Map of the objective function divided into domains according to certain ϵ_1 and ϵ_2 values [9].

That way each domain corresponds to one of 9 possible types of blocks defined by symbols

$$s_1^j(\epsilon_1, \epsilon_2) = \begin{cases} \bar{1}, & \delta_1^j < -\epsilon_2, \\ 0, & -\epsilon_2 \leq \delta_1^j \leq \epsilon_1, \\ 1, & \delta_1^j > \epsilon_1, \end{cases} \quad (16)$$

$$s_2^j(\epsilon_1, \epsilon_2) = \begin{cases} \bar{1}, & \delta_2^j < -\epsilon_1, \\ 0, & -\epsilon_1 \leq \delta_2^j \leq \epsilon_2, \\ 1, & \delta_2^j > \epsilon_2, \end{cases}$$

The resulting set of $s_1^j, s_2^j, j \in [1 : l]$ symbols is used to assess optimization problem features based on the values

of generalized information content function [9]:

$$gic(\epsilon_1, \epsilon_2) = - \sum_{a \neq b} p_{ab}(\epsilon_1, \epsilon_2) \log_6 p_{ab}(\epsilon_1, \epsilon_2). \quad (17)$$

Here $p_{ab}(\epsilon_1, \epsilon_2)$ is the probability assessment of ab type block detection calculated by modified formula (8):

$$p_{ab}(\epsilon_1, \epsilon_2) = \frac{\sum_{k \in N_{ab}} \alpha^k}{\sum_{j=1}^l \alpha^j}, \quad (18)$$

where N_{ab} is the set of indices of ab type blocks, α^j is the angle assumed by $X_{i_2}^j X_{i_1}^j$ and $X_{i_2}^j X_{i_3}^j$ vectors. With the modified formula the impact of any block on $gic(\epsilon_1, \epsilon_2)$ function is weighted by its α^j value. In the case of $\alpha^j = \pi$ for all the blocks we get the original formula (8).

The plot of the $gic(\epsilon_1, \epsilon_2)$ characterizing function is shown in Figure 10.

Figure 10: Plot of $gic(\epsilon_1, \epsilon_2)$ characterizing function for landscape sample of Ackley (a) and Rosenbrock (b) test functions.

4.2 Sectorization method

Let us define a sector of a Variability Map as an area limited by two half-lines with inclination angles $-\pi/4 + \gamma$ and $-\pi/4 - \gamma$ from the δ_1 axis. Here parameter γ defines the size of covered area limited by the radius equivalent to the distance $\max_{j \in [1:l]} \sqrt{(\delta_1^j)^2 + (\delta_2^j)^2}$ between the coordinate center and the remotest point. In Figure 11, the examples of Variability Map sectors corresponding to different values of angle γ are shown.

Let us introduce the characterizing function of *sector density* of a FVM:

$$p(\gamma) = \frac{\sum_{k \in N_\gamma} \alpha^k}{\sum_{j=1}^l \alpha^j}, \quad (19)$$

where N_γ is the set of indices of blocks located in the sector defined by angle γ . The function is monotonously increasing in $\gamma \in [0; \pi]$ range and allows to assess the probability of detecting any block T^j in the given sector. The values of the function are proportional to the number of points within the sector and belong to the $[0; 1]$ range.

Figure 11: FVM sectors that correspond to values $\gamma = \pi/6$ (a) and $\gamma = \pi/2$ (b)

The Sectorization method of FVM analysis follows the same steps as the template method of assessing optimization problem features:

1. collect the landscape sample points $X_i, i \in [1 : n]$ and calculate corresponding objective values f_i
2. build a FVM for the landscape sample using the algorithm provided
3. according to the characterization algorithm, calculate the required values of $p(\gamma)$ function
4. compose the resulting vector C of problem's features based of the values obtained.

Plots of $p(\gamma)$ and $\Delta p(\gamma) = p(\gamma) - p(\gamma - \Delta\gamma)$ functions, where $\Delta\gamma$ is discretization step size, are presented in Figure 12. These examples allow to visually detect the differences in distribution of points among the sectors of the FVMs built for the multi-modal Ackley function and the uni-modal Rosenbrock function. According to the plots, the largest gains of $p(\gamma)$ function for Ackley function are around $\pi/4$ and $3\pi/4$ values of γ parameter. The high concentration of points in these areas of FVM indicates prevalence of blocks of $\bar{1}1$ or $1\bar{1}$ type (in terms of ICoFiS method) which is typical of multi-modal areas of objective function. Same plots for the Rosenbrock function show that the largest increase of the $p(\gamma)$ function accounts around the $\pi/2$ value. That is consistent with the fact that in landscape samples of function with few local minimums the vast majority of blocks are of neutral 11 or $\bar{1}\bar{1}$ types.

Distribution of points among FVM sectors largely depends on such features of objective function that are sig-

Figure 12: Plots of $p(\gamma)$ and $\Delta p(\gamma)$ functions with $\Delta\gamma = 0.01\pi$ for Ackley and Rosenbrock test functions.

nificant for the optimization algorithm (e.g. the function's multi-modality). On the contrary, the distance between FVM points and the coordinate center apparently does not affect values of $p(\gamma)$ function and to a lesser extent reflects key features of objective function's landscape. This distance mostly depends on the growth rate of objective function values in the corresponding area of the design space. Such invariance of the resulting estimates to the scale of δ_1^j, δ_2^j differences potentially improve the generalization ability of the method of optimization problem classification.

5 Efficiency of FVM analysis methods

A FVM analysis method can be the basis for a whole class of characterization algorithms each of which defines its own set of characteristic features. The characterizing function is specified by FVM analysis method, while the components of C vector and its size are set by the characterization algorithm.

Efficiency of a FVM analysis method can be measured by efficiency of characterization algorithm based on it. The latter is determined by efficiency of optimization problems classification based on characteristic features established by that particular algorithm. Efficiency of problems classification is also largely dependent on the sensitivity of the chosen values of the characterizing function both to variations in the landscape sample and to minor changes in objective function landscape.

Any characterization algorithm a_C can be described by:

- size $|C|$ of vector of features,
- characterizing function $g(Y)$ defined by FVM analysis method,
- set of arguments values $\{Y_1, \dots, Y_{|C|}\}$ at which $g(Y)$ function values are calculated and used as C vector components.

One can also use some notable values of arguments Y as components of the vector C , e.g. the value at which $g(Y)$ function reaches its maximum or minimum [21].

5.1 Efficiency metrics of characterization algorithm

Let us formulate basic correctness criteria of characterization algorithm a_C to be assessed by efficiency metrics:

1. values of C vectors obtained for different landscape samples of a given objective function should be similar,
2. values of C vectors obtained for different landscape samples of "similar" objective functions should be similar as well.

Similarity of C vectors can be formalised as a distance (e.g. Euclidean) between corresponding points in characteristic feature space. Notably, metrics for the second criterion can only be set on the basis of expertise providing some insight on objective functions "similarity", e.g. in terms of the way of optimization algorithm functioning. In the simplest case, when dealing with test optimization problems, one can assess multi-modality of objective functions and use the number of local minimums particularly for the comparison. Another example is using labels of B^* strategies that show the best efficiency $e(B, C, I)$ of solving given optimization problems $q(C)$. That way, similarity of objective functions is determined by similarity of the best suited strategies.

To assess efficiency $s(a_C)$ of a characterization algorithm we suggest to use silhouette coefficient [23] of the labeled sample of vectors $C_r, r \in [1 : R]$ calculated by that algorithm for a set of optimization problems:

$$s(a_C) = \frac{1}{R} \sum_{r=1}^R \frac{d_r^{\text{out}}(a_C) - d_r^{\text{in}}(a_C)}{\max(d_r^{\text{out}}(a_C), d_r^{\text{in}}(a_C))} \in [-1; 1], \quad (20)$$

where $d_r^{\text{in}}(a_C)$ is the average distance from C_r vector of a certain cluster to other vectors of the same cluster, $d_r^{\text{out}}(a_C)$ is the average distance from C_r vector of a certain cluster to vectors of the closest cluster. The maximum value of the silhouette coefficient corresponds to a labeled sample with clearly allocated and not overlapping clusters. Silhouette-

based efficiency metrics will assess compactness of clusters of C_r vectors as well as separation of the clusters from each other.

To obtain $s(a_C)$ estimates based on criterion 1, one can simply use the labels h_r corresponding to optimization problems for the landscape samples of which the vectors C_r were calculated. In the case of involving the expert knowledge to measure the correctness by criterion 2, the labels h_r are set in accordance with classes of optimization problems allocated by an expert (e.g. based on the level of multimodality of objective function characterized by C_r vector).

Having efficiency metric $s(a_C)$ defined we can formulate the problem of finding the best characterization algorithm a_C^* that has the maximum efficiency value

$$\max_{a_C \in A_C} s(a_C) = s(a_C^*), \quad (21)$$

where A_C is a set of characterization algorithms based on various FVM analysis methods.

5.2 Efficiency assessment of FVM analysis methods

The following rules were applied for building characterization algorithms based on the considered FVM analysis methods:

- In the case of characterizing function $g(Y)$ is one-dimensional, i.e. $|Y| = 1$, the vector C is composed of the function values in points distributed evenly in its argument domain.
- If characterizing function $g(Y)$ has several arguments, i.e. $|Y| > 1$, its maximum values along each argument, with the others fixed, are used as components of the vector C .

The formulas of vector $C = (c_1, \dots, c_{|C|})$ components are presented below for characterization algorithms a_C based on each considered FVM analysis method:

- ICoFiS method (section 3.2):

$$ic\left(\epsilon_k = \epsilon_{\min} + \frac{k}{|C| - 1}(\epsilon_{\max} - \epsilon_{\min})\right), \quad k \in [0 : |C| - 1] \quad (22)$$

- Generalized Information Content method (section 4.1):

$$\max_{\epsilon_1} gic(\epsilon_1, \epsilon_2 = \epsilon_k), \max_{\epsilon_2} gic(\epsilon_1 = \epsilon_k, \epsilon_2), \quad \epsilon_k \in [\epsilon_{\min}; \epsilon_{\max}], k \in [0 : \frac{|C|}{2} - 1] \quad (23)$$

- Sectorization method (section 4.2):

$$p\left(\gamma_k = \frac{k}{|C| + 1}\pi\right), \quad k \in [1 : |C|] \quad (24)$$

In Figure 13, as an example for Ackley function, the plot of $gic(\epsilon_1, \epsilon_2)$ characterizing function is shown with dots marking $c_1, \dots, c_{16}$ values calculated by formula (23).

Figure 13: Plot of $gic(\epsilon_1, \epsilon_2)$ characterizing function with the dots indicating its values used as components of C vector for a test Ackley function.

The general procedure of estimating the efficiency of characterization algorithm a_C using test optimization problems has the following steps:

1. For each of the test problems generate landscape samples $\mathbb{X}_j = \{X_{i,j}, i \in [1 : n]\}$, $j \in [1 : N]$, each containing n points randomly located in D_X domain.
2. Build a FVM for each landscape sample $\mathbb{X}_j$ and calculate the components of vector C_j in accordance with the characterization algorithm a_C . As a result, there is a set of points C_r , $r \in [1 : R]$ in the feature space ($R = N \times \text{number of test problems}$).
3. Assign h_r labels of a chosen type H to each C_r vector to get a labeled sample $\{(C_1, h_1), \dots, (C_R, h_R)\}$.
4. Calculate the value of $s(a_C)$ metric (i.e. silhouette coefficient) to estimate the quality of the cluster structure of sample $(C_r, h_r)_{r=1}^R$.

The type H of labels is defined by the criterion chosen for efficiency assessment (see 5.1). The labeled sample $(C_r, h_r)_{r=1}^R$ can then be used to build a classification model $\hat{h}(C)$ predicting labels of optimization problems based on landscape sample features $C \in D_C$. The quality of predictions may also provide an indication of the characterization algorithm's efficiency [22]. However, the quality metric

of the cluster structure of the sample $(C_r, h_r)_{r=1}^R$ provides direct assessment of the algorithm efficiency without any additional steps, i.e. building a classification model and evaluating the accuracy of the predicted labels $\hat{h}^r$.

5.3 Computational experiment

In this section, comparative analysis of characterization algorithms' efficiency is conducted following the steps provided in section 5.2. For the visual assessment of the efficiency of various characterization algorithms, algorithms that set a two-dimensional vector of features were also considered.

The following test optimization problems were used for the computational experiment:

- problems q_1, q_2 based on Rosenbrock and Himmelblau test functions respectively with a small number of local minimums (Figure 14a,b).
- problems q_3, q_4 based on Ackley and Rastrigin test functions respectively with a large number of local minimums (Figure 14c,d).

Figure 14: Plots of the test objective functions in the search domain D_X for problems q_1 (a), q_2 (b), q_3 (c) and q_4 (d).

Let us compose two-dimensional vectors $C = (c_1, c_2)$ based on the values of characterizing functions $ic(\epsilon)$, $gic(\epsilon_1, \epsilon_2)$ and $p(\gamma)$ for illustrative purposes:

$$C^{ic} = \left(ic(\epsilon = 0), \max_{\epsilon} ic(\epsilon) \right), \quad (25)$$

$$C^{gic} = \left(\max_{\epsilon_1} gic(\epsilon_1, \epsilon_2 = 0), \max_{\epsilon_2} gic(\epsilon_1 = 0, \epsilon_2) \right), \quad (26)$$

$$C^p = \left(p(\gamma_i = \frac{\pi}{3}), p(\gamma_i = \frac{\pi}{6}) \right). \quad (27)$$

The characterization algorithms setting features given by the formulas are denoted as a_C^{ic} , a_C^{gic} , a_C^p correspondingly. According to the general procedure of estimating the efficiency of a_C algorithm, the following steps were performed for each of a_C^{ic} , a_C^{gic} , a_C^p algorithms:

1. For each of the $q_1 - q_4$ problems generate $N = 100$ landscape samples, each containing $n = 100$ points randomly located in D_X domain.
2. For each landscape sample $\mathbb{X}_r$ build a FVM and calculate the components of vector C_r , $r \in [1 : 400]$.
3. Assign labels h_r of $H^1 = \{h^1, \dots, h^4\}$ type to the vectors C_r , each label indicating the problem for landscape sample of which the vector was calculated.
4. Calculate the value of $s(a_C)$ metric for the resulting sample $(C_r, h_r)_{r=1}^{400}$, $h_r \in H^1$.

In Figure 15, the samples $(C_r, h_r)_{r=1}^{400}$ obtained by a_C^{ic} , a_C^{gic} and a_C^p algorithms are shown with labels $h_r \in H^1$ represented by the colours of points.

Figure 15: Set of C_r , $r = [1 : 400]$ vectors obtained for $q_1 - q_4$ problems by characterization algorithms a_C^{ic} (a), a_C^{gic} (b) and a_C^p (c); colours of points mark the corresponding H^1 labels.

As is shown in Figures 15a,b, in the case of using a_C^{ic} and a_C^{gic} algorithms the groups of C_r points corresponding to $q_1 - q_4$ problems are arranged in the feature space in ascending order of multi-modality degree of their objective functions. In the case of using a_C^p algorithm (see Figure 15c), there is a clear separation of points along the c_1 axis by the multi-modality degree of $q_1 - q_4$ problems: the clusters of points related to q_1 and q_2 problems on the left; the clusters of points related to q_3 and q_4 problems on the right.

The ability of characterization algorithms to separate points in feature space according to the multi-modality level of the corresponding objective functions can also be measured by applying labels of $H^2 = \{h', h''\}$ type to C_r vectors. The C_r vectors characterizing landscape samples of functions with a small number of local minimums (q_1, q_2 problems) are labeled with h' and those characterizing highly multi-modal functions (q_3, q_4 problems) — with h'' . Table 1 contains the values of the $s(a_C)$ metric calculated for a_C^{ic} , a_C^{gic} , a_C^p algorithms with the use of labels of H^1 and H^2 types.

Table 1: Values of $s(a_C)$ metric calculated with different types of labels, $|C| = 2$

Algorithm	Label type	
	H^1	H^2
a_C^{ic}	0.53	0.57
a_C^{gic}	0.55	0.60
a_C^p	0.34	0.67

One can obviously use more features to characterize optimization problems. A proper comparison of FVM analysis methods based on the efficiency of corresponding characterization algorithms requires analysis of feature spaces of higher dimensionality. To get statistically more reliable results we also performed computational experiments with landscape samples of different sizes.

Figure 16 presents the results of the efficiency analysis of characterization algorithms a_C^{ic} , a_C^{gic} , a_C^p in a 64-dimensional feature space. It shows an obvious fact that the more points there are in a landscape sample, the higher the accuracy of estimates made by characterization algorithm and, therefore, the higher its efficiency. According to Figure 16a, the most efficient characterization algorithms in terms of criterion 1 are those based on the Sectorization and Generalized Information Content methods. However, in terms of criterion 2, the efficiency of the algorithms based on the Sectorization method in the case of classifying

Figure 16: Values of $s(a_C)$ metric depending on the size of the landscape sample for a_C^{ic} , a_C^{gic} , a_C^p characterization algorithms with labels of types H^1 (a) and H^2 (b), $|C| = 64$.

ing optimization problems by the multi-modality level of their objective functions is significantly higher than the efficiency of other algorithms (Figure 16b). Using vector of features built of the values of $p(\gamma)$ and $gic(\epsilon_1, \epsilon_2)$ characterizing functions may have a positive impact on the generalizing ability of optimization problem classification algorithm.

6 Conclusion

The paper presents a new Dependency Decomposition approach to optimization algorithm configuration taking into account the amount of computational resources available for solving the problem. According to the approach, computational complexity of the configuration procedure is reduced by modeling the algorithm's efficiency in its parameter space and problem's feature space separately. Assuming that the efficiency of the algorithm may change in uncertainly different manner along the given dimensions, it may become impractical to build and maintain a unite model. The cascade modeling structure allows to combine different approximation techniques to handle such heterogeneous dependency.

For explicit analysis of significant features of optimization problems the notion of objective function's Variability Map is introduced. A method of building a Full Variability Map was suggested that takes into account the most of objective function variations between the neighbouring points of a landscape sample. The Generalized Information Content method was introduced for Full Variability Map analysis with a view to get more accurate and robust estimations of objective function features. The sectorization method of Full Variability Map analysis was also presented allowing to obtain estimations of objective function features that would be more relevant for the optimization

algorithm efficiency, compared to the other methods considered.

The problem of feature vector composition was also addressed in the paper. A properly designed vector of characteristic features provides the foundation for collecting, analyzing and generalizing information on the efficiency of optimization algorithm depending on the properties of the problems being solved. The procedure of comparative effectiveness study of Variability Map analysis methods was introduced on the basis of the efficiency metrics of characterization algorithms corresponding to these methods. The results of the computational experiments performed indicate the high efficiency and relevance of the proposed methods of Variability Map analysis for optimization problem classification.

References

- [1] Belkhir N. et al., Feature based algorithm configuration: A case study with differential evolution, *International Conference on Parallel Problem Solving from Nature*, Springer, Cham, 2016, 156-166
- [2] Eggensperger K., Lindauer M., Hutter F., Pitfalls and best practices in algorithm configuration, *Journal of Artificial Intelligence Research*, 2019, 64, 861-893
- [3] Shan S., Wang G. G., Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions, *Structural and Multidisciplinary Optimization*, 2010, 41(2), 219–241
- [4] Kerschke P., Trautmann H., Comprehensive feature-based landscape analysis of continuous and constrained optimization problems using the R-package flacco, *Applications in Statistical Computing*, Springer, Cham, 2019, 93-123
- [5] Cavazzuti M., Design of experiments, *Optimization Methods*, Springer, Berlin, Heidelberg, 2013, 13–42
- [6] Li G., Aute V., Azarm S., An accumulative error based adaptive design of experiments for offline metamodeling, *Structural and Multidisciplinary Optimization*, 2010, 40(1), 137–155
- [7] Forrester A. I. J., Keane A. J., Recent advances in surrogate-based optimization, *Progress in Aerospace Sciences*, 2009, 45(1-3), 50–79
- [8] Chatterjee A., Siarry P., Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization, *Computers & operations research*, 2006, 33(3), 859-871
- [9] Agasiev T. A., Generalized Information Content based on Variability Map for Exploratory Landscape Analysis of Global Optimization Problems, *Procedia Computer Science*, 2019, 150, 53-60
- [10] Beiranvand V., Hare W., Lucet Y., Best practices for comparing optimization algorithms, *Optimization and Engineering*, 2017, 18(4), 815-848
- [11] Kerschke P., Trautmann H., Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning, *Evolutionary computation*, 2019, 27(1), 99-127
- [12] Karpenko A. P., Sviaadze Z. O., Meta-optimization based on self-organizing map and genetic algorithm, *Optical Memory and Neural Networks*, 2011, 20(4), 279-283
- [13] Branke J., Elomari J. A., Meta-optimization for parameter tuning with a flexible computing budget, *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, ACM, 2012, 1245-1252
- [14] Kerschke P. et al., Automated algorithm selection: Survey and perspectives, *Evolutionary computation*, 2019, 27(1), 3-45
- [15] Momin J., Yang X. S., A literature survey of benchmark functions for global optimization problems, *Journal of Mathematical Modeling and Numerical optimization*, 2013, 4(2), 150–194
- [16] Mersmann O. et al., Exploratory landscape analysis, *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, ACM, 2011, 829–836
- [17] Beham A. et al., Integrating Exploratory Landscape Analysis into Metaheuristic Algorithms, *International Conference on Computer Aided Systems Theory*, Springer, Cham, 2017, 473-480
- [18] Kerschke P. et al., Cell mapping techniques for exploratory landscape analysis, *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, Springer, Cham, 2014, 115–131
- [19] Flamm C. et al., Barrier trees of degenerate landscapes, *Zeitschrift für physikalische chemie*, 2002, 216(2), 155
- [20] Kerschke P. et al., Detecting funnel structures by means of exploratory landscape analysis, *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, ACM, 2015, 265–272
- [21] Muñoz M. A., Kirley M., Halgamuge S. K., Exploratory landscape analysis of continuous space optimization problems using information content, *IEEE Transactions on Evolutionary Computation*, 2015, 19(1), 74–87
- [22] Saini B. S., Lopez-Ibanez M., Miettinen K., Automatic Surrogate Modelling Technique Selection based on Features of Optimization Problems, 2019
- [23] Guerra, Luis, et al., A comparison of clustering quality indices using outliers and noise, *Intelligent Data Analysis*, 2012, 16(4), 703-715