Variable Reduction for Surrogate-Based Optimization

Frederik Rehbach
frederik.rehbach@th-koeln.de
TH Köln
Cologne, Germany

Lorenzo Gentile
lorenzo.gentile@th-koeln.de
TH Köln
Cologne, Germany

Thomas Bartz-Beielstein
thomas.bartz-beielstein@th-koeln.de
TH Köln
Cologne, Germany

ABSTRACT

Real-world problems such as computational fluid dynamics simulations and finite element analyses are computationally expensive. A standard approach to mitigating the high computational expense is Surrogate-Based Optimization (SBO). Yet, due to the high-dimensionality of many simulation problems, SBO is not directly applicable or not efficient. Reducing the dimensionality of the search space is one method to overcome this limitation. In addition to the applicability of SBO, dimensionality reduction enables easier data handling and improved data and model interpretability. Regularization is considered as one state-of-the-art technique for dimensionality reduction. We propose a hybridization approach called Regularized-Surrogate-Optimization (RSO) aimed at overcoming difficulties related to high-dimensionality. It couples standard Kriging-based SBO with regularization techniques. The employed regularization methods are based on three adaptations of the least absolute shrinkage and selection operator (LASSO). In addition, tree-based methods are analyzed as an alternative variable selection method. An extensive study is performed on a set of artificial test functions and two real-world applications: the electrostatic precipitator problem and a multilayered composite design problem. Experiments reveal that RSO requires significantly less time than standard SBO to obtain comparable results. The pros and cons of the RSO approach are discussed, and recommendations for practitioners are presented.

CCS CONCEPTS

• **Mathematics of computing → Discrete optimization**; • **Theory of computation → Continuous optimization**; **Gaussian processes**; • **Computing methodologies → Modeling and simulation**;

KEYWORDS

Dimensionality Reduction, LASSO, Surrogate-Based Optimization, Surrogates, Modeling, Real-World

ACM Reference Format:
Frederik Rehbach, Lorenzo Gentile, and Thomas Bartz-Beielstein. 2020. Variable Reduction for Surrogate-Based Optimization. In *Genetic and Evolutionary Computation Conference (GECCO '20), July 8–12, 2020, Cancún, Mexico.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3377930.3390195

GECCO '20, July 8–12, 2020, Cancún, Mexico

ACM ISBN 978-1-4503-7128-5/20/07…$15.00
https://doi.org/10.1145/3377930.3390195

1 INTRODUCTION

Real-world optimization problems often have multiple characteristics that clearly distinguish them from artificial test-functions.

Mostly these characteristics impede an easy or quick optimization approach. Among others, two of the most concerning criteria are being high dimensional and being costly to evaluate. A popular approach for the optimization of costly functions is Surrogate-Based Optimization (SBO). In SBO a data-driven surrogate model is used as a substitute for the real objective function. While the fitting process requires some computational effort, the fitted model is much cheaper to evaluate than the original function. A so-called infill criterion is used to judge the quality of each candidate solution on the surrogate model. Such a criterion could for example simply be the predicted function value. An internal optimizer, such as an evolutionary algorithm (EA), is used to extensively search for the optimum of the cheap to evaluate surrogate. This optimum of the infill criterion is evaluated on the real objective function leading to an additional information gain that is used for updating and hopefully improving the surrogate model. This process is iteratively repeated.

A more detailed explanation of SBO and its applications can be found in [3] and [23]. Dealing with high dimensional problems usually requires a large number of function evaluations to obtain an adequate model fit. Yet, exactly this largely contradicts with a high cost of function evaluation. Furthermore, even trying to build a surrogate on high-dimensional data causes problems. Firstly, the computational effort related to fitting a model scales with the dimensionality of the data. Fitting a high-dimensional surrogate model might turn out to be just as expensive as evaluating the objective function itself and thus diminishes the usefulness of SBO. Furthermore, depending on the applied model, fitting a problem with high dimensionality might even be completely infeasible. This leads to the first of the two main research questions:
(Q-1) How can SBO efficiently be applied to high-dimensional objective functions?

The second research question stems from ongoing research cooperations with industry. Simply delivering a black-box type algorithm which somehow manages to find good solutions for a given problem is neither ideal nor even accepted. In order to reach acceptance for a proposed new methodology, algorithms have to deliver interpretable results. High-dimensional surrogate models are not well interpretable. Having a model that only uses a few important variables in order to predict a certain function increases the understandability and acceptance of an algorithm in industry. Therefore, the second research question is:
(Q-2) How can the interpretability of high-dimensional SBO be increased?

The rest of this paper is structured as follows: Section 2 gives an overview of related research and introduces methodologies for

variable selection. In Section 3, two applications from industry and a set of artificial test-functions are presented. They are used to judge the performance of our proposed methodology, which is presented in Section 4. Section 5 describes the experimental setup. Results are presented in Section 6 and finally discussed in Section 7. The paper concludes with a short outlook on future work.

2 RELATED RESEARCH

2.1 Kriging and Efficient Global Optimization

One of the most frequented model choices for low-budget optimization is Kriging. Kriging uses Gaussian process models in order to estimate the objective function value of some given candidate solution. The predictions of Kriging are based on a correlation structure which is calculated from the set of all evaluated candidate solutions [9].

Kriging can deliver a high prediction accuracy even if only a very limited dataset is available for the model fitting process. Additionally, Kriging is also favored in SBO, due to its ability to estimate the models uncertainty. Based on some candidate solution, Kriging generates a normal distribution of the corresponding objective value, where the mean is the predicted value and the standard deviation is the uncertainty.

Importantly, Krigings uncertainty estimate is used in the calculation of the Expected Improvement (EI) infill criterion for a given candidate solution [17]. EI is used in the popular Efficient Global Optimization (EGO) algorithm [13]. EGO is based on the idea that instead of searching for the candidate solution with the best predicted objective function value, one can take into account the models uncertainty estimate. The EI criterion combines the benefits of an explorative search (search in unknown regions with high uncertainty) with those of an exploitative search (only search in the best predicted region). Therefore, EGO is known as a well balanced algorithm for global optimization that does not get as easily stuck in local optima.

2.2 Least Absolute Shrinkage and Selection Operator

Regularization techniques are often used in engineering or other applied sciences to face high dimensional regression problems. While a number of different regularization methods are commonly used, the Least Absolute Shrinkage and Selection Operator (LASSO) is very popular. It is often chosen for its efficient variable selection property [27, 28]. This capability helps to deal with problems where the number of variables is larger than the total number of observations, as it shrinks the coefficients of non-important parameters to zero. Therefore, the approach can be used parameter free. The user does, for example, not have to select a specific threshold for variable importance, as all non-important variables are assigned exactly zero. Ordinary least squares (OLS) is the standard technique for estimating the unknown parameters in a linear regression model. In ordinary least squares [8], we estimate the parameters by minimizing the sum of the squared errors:

$$\hat{\beta}^{OLS} := \arg\min_{\beta} \left\{ ||\mathbf{Y} - \mathbf{X}\beta||_2^2 \right\} = \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T \beta)^2 \right\},$$

where n is the number of observations. The basic idea of LASSO is to add a penalty term to the least squares problem, in order to penalize non-zero parameter values. This is done in the following way:

$$\hat{\beta}_\lambda := \arg\min_{\beta} \left\{ ||\mathbf{Y} - \mathbf{X}\beta||_2^2 + \lambda ||\beta||_1 \right\} \quad (1)$$

$$= \arg\min_{\beta} \left\{ \sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}, \quad (2)$$

where $||\beta||_1$ is the so called l_1-penalty and p is the number of decision variables also called *predictors*. It is worth noticing that setting $\lambda = 0$ results in $\hat{\beta}_\lambda \equiv \hat{\beta}^{OLS}$. The choice of λ, also referred to as *regularization* or *penalty* parameter, is crucial. In fact, it handles a trade–off between goodness–of–fit and sparsity. However, finding a good value for λ does not require an additional user determined parameter. λ can be estimated during variable selection by employing cross-validation. Here, a common choice for the penalty value is the one that minimizes cross-validation error.

2.3 Tree-Based Methods: Random Forests

Random forests, which were introduced by Breiman [5], build an ensemble of tree based predictors. Each tree is grown on a randomly sampled (bagged) subset of the training data. Random forests are well-established robust models that can be build to fit mostly any data independent of the class of used variables. Thus, also mixed problems e.g. with numeric, integer, and categorical variables can be fitted. Among others, random forests gained publicity in being used for regression and classification through Liaw et al. [15].

An extremely useful byproduct of random forests is their inherent capability of estimating variable importance. There are multiple possibilities to assess a set of given variables. Arguably the two most used of these are the out-of-bag (OOB) error and the Gini criterion. The OOB error is a bootstrapped methodology that estimates the importance of a variable. The variable is removed as an input for the bagged trees, and the quality drop in the prediction is observed. A high difference in the OOB error is a sign for high variable importance. The second heuristic, the Gini criterion, is calculated by measuring the decrease in the Gini node impurity at each split while building a tree. The average decrease in impurity for a given variable is then assigned as the variables importance [1].

The availability of variable importance measures even for mixed and discrete problems renders random forests to be a good fit for the industry problems which are covered in the next sections. Tutz et al. [29] also make use of this characteristic of tree-based methods. They argue that tree-based approaches outperform generalized linear models in variable importance assessments when it comes to higher-order terms and variable interactions. Yet, tree-based methods tend to neglect the main effects [29].

3 PROBLEMS

3.1 The Electrostatic Precipitator Problem

The electrostatic precipitator (ESP) problem is an industrial optimization problem that is part of ongoing research. A comprehensive

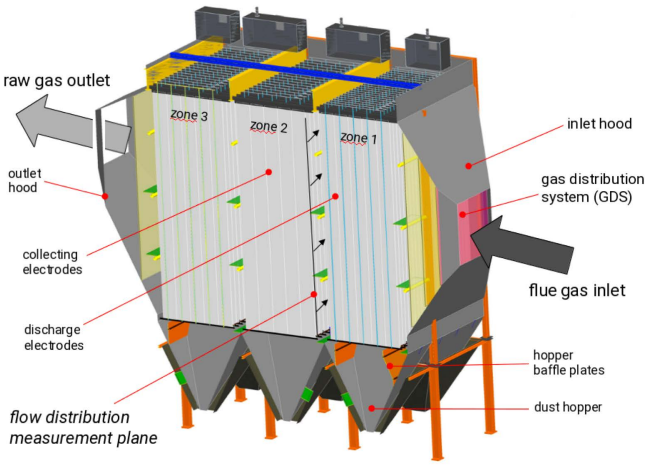


Figure 1: ESP with 3 separation zones. This figure was kindly provided by Steinmüller Babcock Environment GmbH.

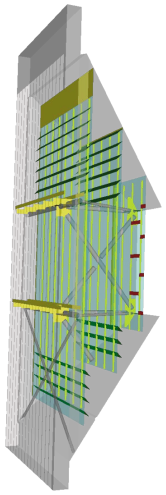


Figure 2: Visualization of a gas distribution system (GDS) mounted in the inlet hood of an ESP. This figure was kindly provided by Steinmüller Babcock Environment GmbH.

introduction to the ESP-problem which is based on [25] is given in the following:

The ESP is one of the main components of gas cleaning systems. They are used in large scale coal-fired power plants or other industries where solid particles have to be removed from a gas stream. Electrostatic precipitators are large devices with dimensions of up to 30m × 30m × 50m, resulting in multiple millions of Euros just in steel building cost. The main task of an ESP is to separate and extract particles from exhaust gases in order to reduce environmental pollution. Figure 1 illustrates this system.

In the flue gas inlet hood of an ESP, a gas distribution system (GDS) (shown in Figure 2) is required to control and guide the gas flow through separation zones in which particles are removed from the exhaust gases. If no GDS is used, or if the system is configured poorly, the fast inlet gas stream will rush through the separation zones of the ESP. This results in very low separation efficiencies. In case of a well configured GDS, the inflowing gas is nicely distributed across the whole surface of the separation zones, resulting in high efficiency. Hence, the efficient operation of an ESP requires an optimal configuration of the GDS.

The GDS in the given industry project consists of 334 slots. Each of these slots can be configured with baffles, as well as blocking and perforated plates. Baffles are metal plates which are mounted at an angle to the general gas flow. They are used to redirect a gas stream into a new direction. Blocking plates completely block a gas stream. Perforated plates are used to slow down and only partially block gas streams. They are created by punching a grid of holes into metal plates. Smaller holes lead to higher pressure drops and thus a slower gas stream. Larger holes allow for a nearly free gas flow. These plates can be mounted into each of the 334 configurable slots. Optionally, some slots can also be left empty allowing the air to pass through.

A single Computational Fluid Dynamics (CFD) simulation of the system requires up to eight hours running in parallel on 16 cores. This renders the models usage as a test-function in the development of new algorithms as infeasible. For this reason, a smaller model with lowered computational cost was created. The resulting model only requires between one and two minutes of runtime compared to the multiple hours of the full model. This speed-up comes at the cost of reduced simulation accuracy. However, the reduced model still captures some of the difficulties and complex features of the actual problem, while enabling a much more detailed experimental study. Reproducing the rugged problem landscape is much more important than the actual accuracy of each sample point.

The reduced model only uses 49 instead of the original 334 configuration slots. Each of the 49 slots can be configured with one of seven different types of plates including leaving the slot empty. The vast amount of possible configurations for the GDS reveals a complex discrete optimization problem. Each single evaluation of a given configuration, still requires to run a computationally expensive CFD simulation thus resulting in large runtimes for experimental studies.

The open source CFD framework OpenFOAM [30] was used to perform our simulations. The original landscape of a real industrial problem is transferred into a function which can be evaluated in reasonable computation time. The ESP problem was therefore considered as a good industrial benchmark for this paper. Currently, the accelerated simulation model is used to tune and advance the research in algorithms for the optimization of the full long-running simulation.

3.2 The Sandwich-Structured Composite Plate Design Problem

Sandwich panels are composites structures consisting of two thin laminate outer skins and lightweight, e.g., honeycomb, thick core structure. Owing to the core structure, such composites are distinguished by stiffness. Despite the thickness of the core, sandwich composites are light and have a relatively high flexural stiffness. Given the considerable mechanical properties and the minimum

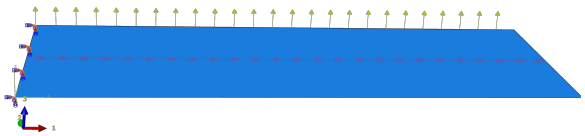


Figure 3: Loads and boundary constrains applied to sandwich-structured composite plate. The yellow arrows represent the applied lifting loads, the purple ones the torquing loads, and the blue and orange ones the encastre boundary conditions.

weight, they represent one of the most attractive solutions in numerous applications in the fields of aeronautics, road vehicles, ships, and civil engineering.

The objective of the Sandwich-Structured Composite Plate Design (SCPD) optimization problem is to target the optimum design of a sandwich-structured composite plate. The plate has been loaded by lifting and torquing loads, linearly distributed along the length of the plate. The lifting loads are applied on the nodes lying in the one of the edges and the torquing ones on the nodes in the centerline. An encastre at the root of the plate has been enforced as shown in Fig. 3.

The objective of the optimization is then to minimize the displacement in the out-of-plane direction of one of the vertices of the plate at the free edge. For accurately computing this, the finite element analysis solver Abaqus [11] has been employed.

In the SCPD problem, the plate has been modeled with 19 different layers of equal thickness but different materials. The 13 central ones, the core, are constituted by Al-hexagonal-honeycomb. The two thin laminate outer skins are made by one layer of fiber reinforced Carbon-Epoxy composite (outer) and two layers of fiber reinforced Kevlar-Epoxy(inner). The adopted materials and their properties are reported in Table 1. As one can see, the core is constituted by an isotropic materials and the skins by orthotropic materials. In case of orthotropic materials, the stiffness of the material is crucially affected by the lamination angle. Contrarily, isotropic materials have equal in-plane and out-of-plane Young's modulus. Hence, the lamination angle of isotropic materials does not affect the material behavior.

In this problem, the lamination angles defining the design have been considered as the decision variables of the optimization. It is then clear that all the variables describing the lamination of the layers made by Al-honeycomb have no influence in the response.

The purpose of this test case is to show the effective improvements given by integrating the dimensionality reduction in the SBO process.

3.3 Artificial Problems

In addition to the two discussed real-world problems, a multitude of artificial test-functions is utilized in this study. Since these functions are cheap to evaluate, a surrogate model-based approach for their optimization is not efficient. However, we argue that the results obtained by optimizing these functions are transferable to other more costly functions. The functions stem from a popular collection

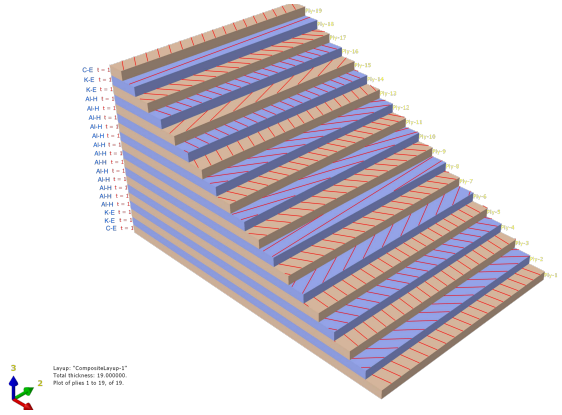


Figure 4: Ply stack configurations. For each ply the thickness, identical for all the plies, the lamination angle and the constituent material are depicted. The red lines represent the lamination angles.

Table 1: Properties of the Material used in the study of the sandwich-structured composite plate design problem.

Properties	Young's modulus 0	Young's modulus 90	In-plane Shear modulus	Poisson's Ratio
Symbols	E1	E2	G12	ν 12
Units	GPa	GPa	GPa	
HC-Al	2.03	2.03	4.8	0
Kevlar-Epoxy	41	10.4	4.3	0.28
Carbon-Epoxy	147	10.3	7	0.27

of test functions, they were chosen for their variable importance properties [1]. All chosen test-functions have a subset of important variables as well as a set of variables which is completely irrelevant for the optimization. This simulates the circumstances that are often met in real-world problems quite well. Here, it is often not known which variables impact a system the most, or even at all. There, one can in a lucky scenario estimate the importance of some variables through the experience of a practitioner. Yet very often, the importance of the variables is completely unknown, e.g. like in the ESP-Problem. Table 2 gives an overview on all test-functions which were used in this study. They range from ten dimensions (the two functions given by Linkletter et al. [16]) up to the 30-dimensional function from Morris et al. [19]. Moreover, the portion of important to completely irrelevant variables varies from function to function. The four given functions by Moon [18] are mostly the same function yet with varying difficulty levels to distinguish the five most important variables from other less important ones. The R implementations for each of the artificial test-functions was acquired from the Virtual Library of Simulation Experiments [26]. More details on each respective test-function can be found in the literature cited in Table 2.

[1] https://www.sfu.ca/ ssurjano/screen.html

Table 2: Test-functions and their dimensionality (nDim), the amount of very important variables (nImp), the total amount of relevant variables (nRelevant), and the amount of irrelevant variables (nIrr). Since the real importances for the ESP-Problem are unknown, question marks are used to fill the gaps.

Function Name	nDim	nImp	nRelevant	nIrr
linketal06dec [16]	10	2	6	4
linketal06sin [16]	10	2	2	8
moon10hd [18]	20	5	15	0
moon10hdc1 [18]	20	5	5	15
moon10hdc2 [18]	20	5	15	0
moon10hdc3 [18]	20	5	15	0
oakoh04 [20]	15	5	10	5
morretal06 [19]	30	2	2	28
ESP-Problem	49	?	?	?
SCPD-Problem	19	2	6	13

4 ALGORITHMS

4.1 Surrogate-Based Optimization

The SBO algorithm applied in this study is a standard implementation based on EGO as described in 2.1. Thus, a Kriging surrogate is iteratively fitted to all available data of the objective function. The expected improvement infill criterion is optimized on the surrogate in order to propose a next candidate solution. The R implementation in the 'CEGO'-package [31] was used for Kriging.

4.2 Regularized Surrogate Optimization

In order to apply SBO efficiently to high-dimensional objective functions, we propose a hybrid algorithm: Regularized Surrogate Optimization (RSO). It consists of standard SBO coupled with an additional regularization stage. The implementation of RSO is explained in Algorithm 1, as well as visualized in Figure 5. During the startup phase of the algorithm, an initial design of candidate solutions is generated and evaluated on the objective function. Instead of now building the surrogate model directly on this data, in RSO a regularization method is used to decrease the dimensionality of the given data. In this study, various methods are compared: three approaches based on LASSO and one based on random forest. The study is programmed in R [24].

As described in Section 2.2, LASSO is usually used for assigning importances to the variables (dimensions) of a given dataset. Yet, due to its characteristic of being able to set the importances of variables to exactly zero, LASSO can easily be transformed into a feature selection method.

Three methods based on LASSO regularization have been tested in this study. They differ on taking or not into account quadratic effects. The first approach is based on the *adaptive* LASSO [32] and aims at identifying only linear effects. It is essentially a two step regularization process that takes advantage of a preliminary Ridge regression [12] to have a clearer distinction between active and inactive variables out of the LASSO regularization. This is done in the following way:

$$\hat{w}_\lambda := \arg\min_\beta \left\{ ||\mathrm{Y} - \mathrm{X}\beta||_2^2 + \lambda_w ||\beta||_2 \right\}$$

$$\hat{\beta}_{\lambda,\lambda_w} := \arg\min_\beta \left\{ ||\mathrm{Y} - \mathrm{X}\beta||_2^2 + \lambda \hat{w}_{\lambda_w} ||\beta||_1 \right\}$$

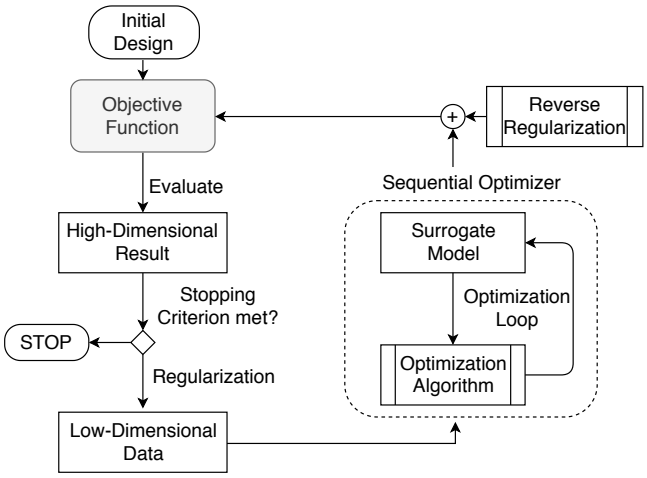


Figure 5: Implementation of the RSO algorithm: After the evaluation of an initial design, the resulting data is reduced through regularization. The lower-dimensional data is fitted with a surrogate model. An optimization algorithm searches for the best candidate solution on the surrogate model. The proposed candidate is merged with information from a reverse regularization method, back to the original dimensionality. The candidate is evaluated and the process is iterated until some stopping criterion is met.

Algorithm 1 Implementation of the RSO algorithm: Here, initDesign() is a function that produces an initial set of candidate solutions, buildSurrogate() is a procedure to build a surrogate model, and optimize() is a suitable optimization algorithm. The function regularization() selects, and reduces $\boldsymbol{X}$ to its important features. revRegularization() refills the missing dimensions of $\boldsymbol{X}$ before eval() can be used to evaluate the new candidate solution on the objective function.

function RSO(initDesign(), buildSurrogate(), optimize(), regularization(), revRegularization(), eval())
  $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ = initDesign()
  $\boldsymbol{y}$ = eval($\boldsymbol{X}$)
  while !(stopping criterion) **do**
    $\boldsymbol{X}_{rd}$ = regularization($\boldsymbol{X}$) ▷ reduce dimensionality of $\boldsymbol{X}$
    model = buildSurrogate($\boldsymbol{X}_{rd}, \boldsymbol{y}$)
    $\boldsymbol{x}_{new}$ = optimize(model) ▷ search optimum of model
    $\boldsymbol{x}_{new}$ = revRegularization($\boldsymbol{x}_{new}$)
    $\boldsymbol{y}_{new}$ = eval($\boldsymbol{x}_{new}$) ▷ evaluate infilled candidate
    $\boldsymbol{X} = \{\boldsymbol{X}, \boldsymbol{x}_{new}\}$
    $\boldsymbol{y} = \{\boldsymbol{y}, \boldsymbol{y}_{new}\}$
  end while
end function

10-fold cross-validation is used to determine both the regressions λ and λ_w values yielding the lowest mean absolute error. The same procedure is also used in the other methods based on LASSO regularization.

The second approach aims at identifying not only linear effects but also *pure* quadratic ones. This consists in a standard LASSO regression where pure quadratic terms are included as predictors.

$$\mathrm{X} = \begin{bmatrix} \mathrm{X} & \mathrm{X} \odot \mathrm{X} \end{bmatrix}$$

The predictors having the coefficients beyond the 90-th percentile are considered as active(*Active*):

$$\begin{aligned} Active_L &= \left\{ i \in [1, n_{var}] : \hat{\beta}_i > q_{0.9}(\hat{\beta}) \right\} \\ Active_Q &= \left\{ i \in [1, n_{var}] : \hat{\beta}_{i+n_{var}} > q_{0.9}(\hat{\beta}) \right\} \\ Active &= Active_L \cup Active_Q \ , \end{aligned}$$

where n_{var} is the number of variables, $Active_L$ and $Active_Q$ are the sets of variables having respectively a considerable linear and pure quadratic effect. For both these firsts two methods the 'glmnet'-package [10] has been adopted to perform the regression.

The third and more ambitious method is intended to screen the active variables on the basis of all the linear and pairwise interactions. This makes use of the 'hiernet'-package [4]. This approach uses two criteria to screen the variables on the basis of their linear and quadratic contribution. It identifies the variables that have a significant linear contribution ($Active_L$) using the same approach used in the previous method. Following the notation of [4], the variables having a strong linear contribution are computed as follows:

$$\begin{aligned} \hat{\beta} &= |\hat{\beta}^+ - \hat{\beta}^-| \\ Active_L &= \left\{ i \in [1, n_{pred}] : \hat{\beta}_i > q_{0.9}(\hat{\beta}) \right\} \ , \end{aligned}$$

To screen all the coefficients related to second-order interactions a clustering approach based on k-means clustering is used. All the coefficients with value exactly zero are removed. Next two clusters are defined using k-means clustering technique. Then the cluster with the greatest mean is identified and selected. Finally, all the variables involved in at least one interaction coefficient are elected as significant variables.

$$\begin{aligned} \mathrm{S}^* &= \arg\min_{\mathrm{S}} \sum_{i=1}^{2} \sum_{\hat{\Theta} \in S_i} ||\hat{\Theta} - \hat{\mu}_i||^2 \\ i^* &= \{ i \in [1, 2] : \hat{\mu}_i = \max(\hat{\mu}) \} \\ Active_Q &= \left\{ j \cup k : \forall \Theta_{j,k} \in S^*_{i^*} \right\} \ , \end{aligned}$$

where $\hat{mu}_i$ is the mean of points in S_i and $\hat{\Theta}$ are the coefficients related to the interactions [4]. The overall set of active variables is then defined as $Active = Active_L \cup Active_Q$.

Variable selection in random forests can be applied in a similar fashion. The algorithms used in this study were implemented with the help of the 'randomForest'-package [15]. Again, cross-validation is used to sequentially remove variables from the dataset in the order of their estimated importance. The set of variables with the lowest cross-validation error is chosen for the next stage.

Next, a surrogate model (in this case, the Kriging implementation of the 'CEGO'-package [31]) is trained. By only fitting the model to the regularized dataset, the time required for the model building is reduced significantly. Just like in standard SBO, some infill criterion can be optimized on the surrogate in order to propose a new candidate solution. The expected improvement criterion, which was described in Section 2.1, was chosen for this process. The new candidate solution cannot be directly evaluated on the objective function. Since the model and thus also the optimizer only searches for the best configuration for the selected variables, a reverse regularization method is required to assign values to the neglected variables. This method assigns a value to each of the variables which are not included in the surrogate-model. Preliminary experiments showed that uniform random sampling outperformed other infill methods. Yet, a more detailed comparison of these reverse regularization methods is out of the scope of this paper.

The combination of the information generated by the reverse regularization method and the optimization of the surrogate model can then be evaluated on the objective function. This process is iteratively repeated until the budget is depleted or some stopping criterion is met.

5 DESCRIPTION OF EXPERIMENTS

5.1 Replicability

In order to assure reproducability, all algorithm and experimenting codes that are described in the following will be published. This sadly excludes the codes to run the ESP-problem as they partially contain company information. All other codes are available on GitHub[2].

5.2 Performance Criteria

Choosing the correct performance criteria for a set of experiments often depends on the cost of each objective function evaluation. On the one hand, if each evaluation is considered cheap, the amount of function evaluations can be ignored and the overall required runtime of the optimization can be measured. On the other hand, the governing factor of cost might be to evaluate the objective function. In that case, one can often neglect the runtime of the optimizers themselves. Instead, there might only be a specific budget for the amount of objective function evaluations that can be done. For example, if one function evaluation is linked to some real-world experiment, a company might only want to spend money for a maximum of 200 such experiments.

Thus, how well an algorithm performs on a certain problem is affected by both algorithm complexity and the problem cost. In order to judge the presented methods from multiple standpoints regarding a varying cost for each objective function evaluation, both of these performance measures are implemented. In each optimization run, the time required to fit and optimize the surrogate model is recorded for each iteration. From this, the total runtime of the optimizer itself can be approximated. Additionally, the best-found objective function value of each run of the algorithms is stored.

By combining these measurements, it is possible to judge which algorithm performs best, considering a given problem cost. On cheap problems, the faster algorithms should outperform the more

[2] https://github.com/frehbach/rehb20b

complex ones. On costly problems, the more efficient algorithms in terms of function evaluations should win.

5.3 Experiment Design and Budget Allocation

Each optimization run starts with a rather small initial design size of 5 candidate solutions, randomly sampled from the design space. Since both of the described real-world problems are expensive simulation tasks, only a small budget is feasible. Therefore, each algorithm is given a total budget of 200 function evaluations. The same budget is kept for the artificial test-functions, even though they would be very cheap to evaluate. This is done in order to improve comparability, as they are meant to simulate expensive to evaluate functions.

In order to optimize the infill criteria of the fitted surrogate models, additional optimization algorithms are necessary. For the real-world problems with mixed-integer variables a mixed-integer evolution strategy [14] from the "CEGO"-package [31] is employed. The Differential Evolution [22] implementation of the "SPOT"-package [2] is used to optimize the surrogates of the artificial test-functions. In both cases, the algorithms were given a budget of 2000 cheap function evaluations to optimize the surrogate infill criterion.

To account for the stochasticity of all algorithms, a number of repetitions are required for each experiment. Yet, numerous experiments on more expensive functions would lead to a very high computational cost. Each algorithm is applied 20 times to each test-function to keep the experiment runtime feasible.

The results of the experiments are examined with a statistical analysis. The same analysis procedure is executed one by one on each test-function. First, a statistical multiple-comparisons test is done. We judge differences between two results to be significant if the given p-values are smaller than $\alpha = 0.05$. A ranking is created by the following procedure: The set of algorithms which is never outperformed with statistical significance is assigned rank one (the best rank). These algorithms are then removed from the list of all algorithms and the same check is performed for the remaining algorithms that are now competing for rank two and so on. This is iteratively continued until all algorithms are assigned with a rank. A pairwise-multiple comparison test, according to Conover [6, 7] was used to check for differences between each of the algorithm pairs. The tests use the R implementations of the "PMCMR"-package [21].

The testing methodology was chosen because the collected data is not normally distributed, and is also heteroscedastic (i.e., group variances are not equal). Therefore, parametric test procedures that require normal distributed data are infeasible. Yet, also non-parametric tests are not free of assumptions. The test assumes the given data stems from statistically independent random samples within and between the groups. Additionally, they are required to have an ordinal measurement scale [6, p. 289]. For the collected optimization results that we consider, these assumptions should hold.

6 RESULTS

The two main objectives of RSO are performance enhancements and runtime reduction. How well these goals were achieved can be observed in Figure 6. For both problems, the results indicate that switching to the RSO approach, regardless of the method adopted, results in a significant decrease in required algorithm runtime with a comparable or even better final objective function result. For example, on the ESP-Problem a more than 5x speedup can be observed.

The differences between the RSO approaches can be compared with the results depicted in Figures 6 and 7, where the results obtained on two artificial test-functions are shown. Specifically, linketal06dec is linearly dependent on all *important* and *relevant* variables. This enhances the performances of Regularized-Surrogate-Optimization (RSO) when adopting the regularization approach based on the Adaptive LASSO or Random Forest and degrades them when using regularization methods that aim at catching non-linear dependencies. In particular, it is interesting to see from Figure 7 that RSO-LASSO-1 is able to screen all active variables from the beginning on, throughout the whole optimization. RSO-LASSO-2 and RSO-LASSO-3 are less able to catch the relevant variables. RSO-LASSO-3 even neglects an important variable on liketal06dec. Contrary, in case of moon10hdc1, the impact of the active variables is completely due to quadratic terms. Therefore, RSO-LASSO-3 results to be the most appropriated. In fact, it rapidly screens the important variables delivering statistically significant better results compared to standard SBO in less than one-fourth of the time.

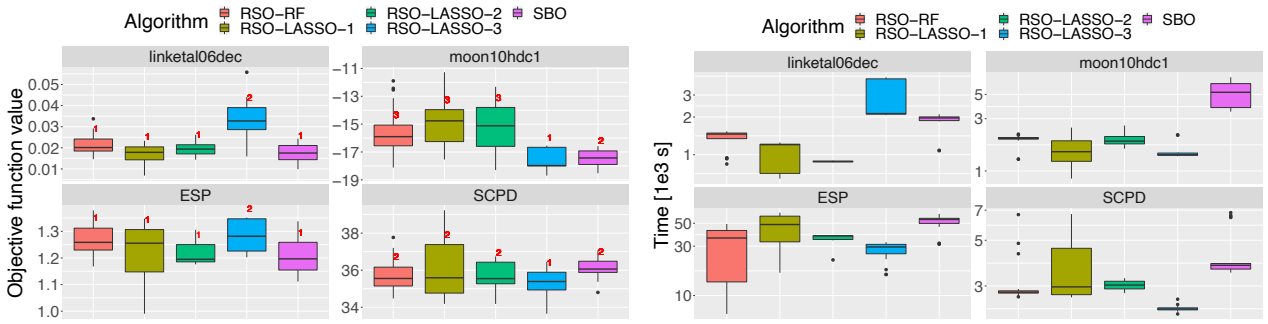
An overview of the statistical analysis of the results can be found in Table 3. A review of the results on the artificial test-functions shows two key observations: Firstly, on the real-world problems, the RSO variants largely deliver the same results in terms of resulting objective function value. On the SCPD problem, RSO-LASSO-3 is even able to outperform standard SBO significantly. Secondly, the artificial objective functions need to be discussed. Here, SBO is never outperformed in terms of resulting objective function quality. Yet, the different RSO approaches often are able to deliver similar results or slightly worse results in less computational time. When there is a statistical quality difference between standard SBO and the RSO approach, then often RSO based on LASSO regularization delivered better quality results than RSO-RF.

Table 3: Performance overview of all algorithms on each test-function. Numbers shown are determined by rank-based pairwise multiple-comparison tests regarding the objective function value. The best result on each problem is marked in bold. The last-row shows the mean rank of each algorithm.

Problem	RSO-RF	RSO-LASSO-1	RSO-LASSO-2	RSO-LASSO-3	SBO
linketal06dec	**1.0**	**1.0**	**1.0**	2.0	**1.0**
linketal06sin	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
moon10hd	2.0	**1.0**	4.0	3.0	**1.0**
moon10hdc1	2.0	2.0	2.0	**1.0**	**1.0**
moon10hdc2	2.0	**1.0**	2.0	2.0	**1.0**
moon10hdc3	3.0	2.0	3.0	2.0	**1.0**
oakoh04	2.0	2.0	3.0	**1.0**	**1.0**
morretal06	2.0	2.0	**1.0**	2.0	**1.0**
ESP-Problem	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
SCPD-Problem	**1.0**	**1.0**	**1.0**	**1.0**	2.0
MeanRank	1.7	1.4	1.9	1.6	**1.1**

7 CONCLUSION AND OUTLOOK

Before drawing a final conclusion and discussing future work, we will reconsider the discussed research questions:



(a) Algorithm comparison of the best objective function value found

(b) Algorithm comparison of runtime

Figure 6: Comparison of all algorithms on two significant test functions and both of the real-world problems. Lower values are better.

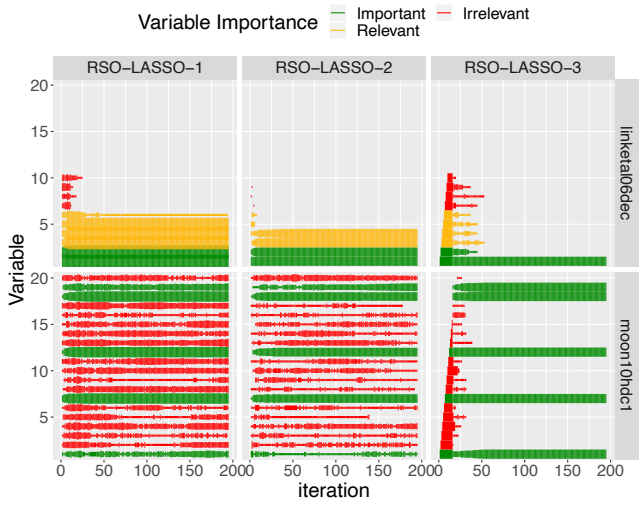


Figure 7: Overview of estimated variable importance by the three RSO-LASSO variations for each iteration. The artificial functions have important (—), relevant (—), and irrelevant (—) terms. The goal is to correctly identify the important and relevant variables while excluding the irrelevant ones. Thus a perfect result does only show yellow or green lines but no red ones. The thickness shows the frequency of the indicated estimation.

The initial research question **(Q-1)** discussed in Section 1 asks whether or not it is possible to enable surrogate-based optimization for high-dimensional problems. The main stated issues were increased modeling time and furthermore even possible model failure due to very high dimensionality.

The results given in Section 6 show that the proposed hybrid algorithm Regularized-Surrogate-Optimization requires significantly less computation time. At the same time it delivers, dependent on the respective optimization problem, comparable results. Most importantly, RSO is applicable even to very high-dimensional problems without the same risk of modeling failure that one would have compared to standard Surrogate-Based Optimization. The reduced modeling run-time can make the RSO algorithm feasible in situations where SBO would be a bad choice due to time constraints.

The second research question **(Q-2)**, concerns about the interpretability of SBO for industry partners. It is often hard to suggest models or algorithms in industry if they are acting in a black-box manner. Thus, if the model itself is based on a complex high-dimensional system, gaining support from field engineers for a new optimization technique is hard. In contrast, a model that is able to identify the most important variables correctly is easier to explain. The selected variables should mostly coincide with the practitioners' field knowledge, confirming his understanding of a system and making it easy to gain support for the new algorithm implementation. The RSO algorithm delivers exactly this interpretability benefit for industry partners. Additionally, it helps in not only delivering the answer of *what* is the best possible configuration for some system. It answers the *why* is this configuration so good, resulting in a further knowledge gain for the company. Since the RSO approach is intrinsically scalable to a specific dimensionality, it is also possible to optimize a specific amount of most important variables of a system without knowing which ones these will be a priori. For example, RSO could, to some extent, combine an initial system analysis with a later optimization routine in time-constrained projects.

These initial results of the application of RSO on industry problems give much room for open research questions. Firstly, the RSO approach is coupled with Kriging. While Kriging is one of the most popular choices in the field of low-budget surrogate optimization, generalizing the algorithm to other surrogates would increase its applicability to more problems. Furthermore, tests on noisy functions or multi-objective functions should be considered. Also, the parameterization of LASSO was evaluated through smaller preliminary experiments. More extensive studies are required for future analysis.

ACKNOWLEDGMENTS

This work was funded by the European Commission's H2020 programme, through the H2020-MSCA-ITN-2016 UTOPIAE Marie Curie Innovative Training Network grant agreement No.: 722734.

REFERENCES

[1] Kellie J Archer and Ryan V Kimes. 2008. Empirical characterization of random forest variable importance measures. *Computational Statistics & Data Analysis* 52, 4 (2008), 2249–2260.

[2] Thomas Bartz-Beielstein, Christian Lasarczyk, and Mike Preuss. 2005. Sequential Parameter Optimization. In *Proceedings Congress on Evolutionary Computation 2005 (CEC'05)*. Edinburgh, Scotland, 1553. http://www.spotseven.de/wp-content/papercite-data/pdf/blp05.pdf

[3] Thomas Bartz-Beielstein and Martin Zaefferer. 2017. Model-based methods for continuous and discrete global optimization. *Applied Soft Computing* 55 (2017), 154 – 167. https://doi.org/10.1016/j.asoc.2017.01.039

[4] Jacob Bien, Jonathan Taylor, and Robert Tibshirani. 2013. A lasso for hierarchical interactions. *Annals of statistics* 41, 3 (2013), 1111.

[5] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[6] William Jay Conover. 1999. *Practical Nonparametric Statistics, 3rd Edition*. Wiley.

[7] William Jay Conover and Ronald L. Iman. 1979. *On multiple-comparisons procedures*. Technical Report Tech. Rep. LA-7677-MS. Los Alamos Sci. Lab.

[8] Norman R Draper and Harry Smith. 1998. Fitting a straight line by least squares. *Applied regression analysis* (1998), 15–46.

[9] Alexander Forrester, Andy Keane, et al. 2008. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons.

[10] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 2010. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software* 33, 1 (2010), 1.

[11] K Hibbitt. 2013. ABAQUS: User's Manual: Version 6.13: Hibbitt. *Karlsson & Sorensen, Incorporated* (2013).

[12] Arthur E Hoerl and Robert W Kennard. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12, 1 (1970), 55–67.

[13] Donald R. Jones, Matthias Schonlau, and William J. Welch. 1998. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization* 13, 4 (01 Dec 1998), 455–492. https://doi.org/10.1023/A:1008306431147

[14] Rui Li, Michael TM Emmerich, Jeroen Eggermont, Thomas Bäck, Martin Schütz, Jouke Dijkstra, and Johan HC Reiber. 2013. Mixed integer evolution strategies for parameter optimization. *Evolutionary computation* 21, 1 (2013), 29–64.

[15] Andy Liaw, Matthew Wiener, et al. 2002. Classification and regression by randomForest. *R news* 2, 3 (2002), 18–22.

[16] Crystal Linkletter, Derek Bingham, Nicholas Hengartner, David Higdon, and Kenny Q Ye. 2006. Variable selection for Gaussian process models in computer experiments. *Technometrics* 48, 4 (2006), 478–490.

[17] Jonas Mockus, Vytautas Tiesis, and Antanas Zilinskas. 1978. *Towards Global Optimization 2*. North-Holland, Chapter The application of Bayesian methods for seeking the extremum, 117–129.

[18] Hyejung Moon. 2010. *Design and analysis of computer experiments for screening input variables*. Ph.D. Dissertation. The Ohio State University.

[19] Max D Morris, Leslie M Moore, and Michael D McKay. 2006. Sampling plans based on balanced incomplete block designs for evaluating the importance of computer model inputs. *Journal of Statistical Planning and Inference* 136, 9 (2006), 3203–3220.

[20] Jeremy E Oakley and Anthony O'Hagan. 2004. Probabilistic sensitivity analysis of complex models: a Bayesian approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 66, 3 (2004), 751–769.

[21] Thorsten Pohlert. 2014. The Pairwise Multiple Comparison of Mean Ranks Package (PMCMR). (2014). http://CRAN.R-project.org/package=PMCMR Accessed on Jan. 12, 2016.

[22] Kenneth Price, Rainer M Storn, and Jouni A Lampinen. 2006. *Differential evolution: a practical approach to global optimization*. Springer Science & Business Media.

[23] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker. 2005. Surrogate-based analysis and optimization. *Progress in aerospace sciences* (2005).

[24] R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/

[25] Frederik Rehbach, Martin Zaefferer, Joerg Stork, and Thomas Bartz-Beielstein. 2018. Comparison of Parallel Surrogate-Assisted Optimization Approaches. In *GECCO '18: Genetic and Evolutionary Computation Conference*.

[26] S. Surjanovic and D. Bingham. 2013. Virtual Library of Simulation Experiments: Test Functions and Datasets. Retrieved February 5, 2019, from http://www.sfu.ca/~ssurjano. (2013).

[27] Robert Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.

[28] Robert Tibshirani. 2011. Regression shrinkage and selection via the lasso: a retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73, 3 (2011), 273–282.

[29] Gerhard Tutz and Moritz Berger. 2015. Tree-structured modelling of categorical predictors in regression. *arXiv preprint arXiv:1504.04700* (2015).

[30] Henry G Weller, G Tabor, Hrvoje Jasak, and C Fureby. 1998. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in physics* 12, 6 (1998), 620–631.

[31] Martin Zaefferer, Joerg Stork, Martina Friese, Andreas Fischbach, Boris Naujoks, and Thomas Bartz-Beielstein. 2014. Efficient Global Optimization for Combinatorial Problems. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation (GECCO'14)*. ACM, New York, NY, USA, 871–878. http://doi.acm.org/10.1145/2576768.2598282

[32] Hui Zou. 2006. The adaptive lasso and its oracle properties. *Journal of the American statistical association* 101, 476 (2006), 1418–1429.