Pre-print: to appear at PPSN 2018
The final authenticated version is available online at https://doi.org/[insert DOI]. (DOI not yet available)

A First Analysis of Kernels for Kriging-based Optimization in Hierarchical Search Spaces

Martin Zaefferer[1] and Daniel Horn[2]

[1] TH Köln, Institute of Data Science, Engineering, and Analytics, Steinmüllerallee 6, 51643 Gummersbach, Germany, martin.zaefferer@th-koeln.de
[2] Technische Universität Dortmund, Faculty of Statistics, Vogelpothsweg 87, 44227 Dortmund, Germany, daniel.horn@tu-dortmund.de

Abstract. Many real-world optimization problems require significant resources for objective function evaluations. This is a challenge to evolutionary algorithms, as it limits the number of available evaluations. One solution are surrogate models, which replace the expensive objective.
A particular issue in this context are hierarchical variables. Hierarchical variables only influence the objective function if other variables satisfy some condition. We study how this kind of hierarchical structure can be integrated into the model based optimization framework. We discuss an existing kernel and propose alternatives. An artificial test function is used to investigate how different kernels and assumptions affect model quality and search performance.

Keywords: surrogate model based optimization, hierarchical search spaces, conditional variables, kernel

1 Introduction

When objective function evaluations become expensive, surrogate models may be employed to reduce the resource consumption in an optimization process. One challenging issue in this context are *conditional* or *hierarchical* variables. Hierarchical variables are only active (i.e., have an influence on the result) if other variables fulfill certain conditions. This occurs in many algorithm tuning problems. For instance, in machine learning algorithms, parameters of an SVM kernel are only active if that kernel is utilized. Similarly, a variable of a variation operator in an evolutionary algorithm only has an effect if that operator is actually used. Such parameters may also occur in engineering problems. For instance, if a variable defining the amount of energy fed into the system exceeds a certain level, it may require an additional cooling step which itself has variables.

We require tools to model these cases efficiently. In previous studies, three alternatives have been employed: Firstly, the hierarchical nature of a variable could be ignored and the data handled by standard modeling methods. This approach could be suboptimal since the available information on variable activity is not used. Secondly, a pre-processing step could impute a constant value for

the inactive variables, e.g., the mean, or some lower/upper bound [1–3]. We refer to this as the *imputation approach*. Thirdly, the information about hierarchical variables can be incorporated into the modeling process. It can be be integrated into the kernel, e.g., the Arc-kernel [4, 5]. In other approaches, Gaussian processes are placed on the leaves of a tree structure that is assumed to represent the hierarchical dependencies of the variables [6–8].

In this article, we focus on kernels in the context of the third case, and propose alternatives to the Arc-kernel. We present a numerical comparison based on a simple test function to verify that the performance of these kernels meets our expectations. We aim to answer the following research questions:

1. Do kernels have to incorporate knowledge about the search space hierarchy?
2. When should which kernel be used?
3. Does definiteness of the kernel play a role?

We give a short introduction to model based optimization in Sec. 2 and to Kriging models in Sec. 3. Afterwards, we introduce kernels for hierarchical search spaces in Sec. 4. We describe our experimental setup in Sec. 5 and analyze the results in Sec. 6. A final evaluation and outlook on future work is given in Sec. 7.

2 Surrogate Model-Based Optimization

Let $f : \mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_d \to \mathbb{R}$ be a black-box function with a d-dimensional input domain and a deterministic output y. Each $\mathcal{X}_i$ can either be numeric and bounded ($\mathcal{X}_i = [l_i, u_i] \subset \mathbb{R}$) or categorical. We want to solve the optimization problem (OP) and find the input $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x})$. We assume that evaluations of f are expensive, which limits the number of evaluations severely.

Sequential model-based optimization (SMBO) is a state-of-the-art method for solving expensive OPs. It is based on the Efficient Global Optimization (EGO) procedure [9]. First, SMBO samples and evaluates an initial set of candidate solutions. Then, a surrogate regression model is fitted to the data. The model is optimized with respect to an infill criterion in order to find a new, promising candidate $\boldsymbol{x}^*$. The candidate $\boldsymbol{x}^*$ is evaluated with f and added to the data set. This allows to train an improved surrogate model. The procedure iterates until a stopping criterion is reached, e.g., a budget on the number of function evaluations. A detailed introduction is given by Bischl et al. in [10].

Four components of the SMBO procedure have to be specified: the generation of the initial candidate set, the surrogate model, the infill criterion and the optimizer of the infill criterion. We use Latin Hypercube Sampling (LHS), Kriging models, the expected improvement criterion and Differential Evolution (DE) [11]. Our methods can be easily extended to other SMBO variants that employ kernel-based models.

3 Kriging

Frequently, SMBO employs Kriging models, which interpret observations as realizations of a Gaussian process. Forrester et al. [12] give a detailed description.

In its core, Kriging models the correlation between observations, e.g., with an exponential correlation function $\mathrm{k}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\theta \cdot \mathrm{d}(\boldsymbol{x}, \boldsymbol{x}'))$. Here, $\boldsymbol{x}$ and $\boldsymbol{x}'$ are samples, θ is a kernel parameter and $\mathrm{d}(\boldsymbol{x}, \boldsymbol{x}')$ is a distance function, e.g., the Euclidean distance if $\boldsymbol{x}$ is real valued. The correlation matrix $\boldsymbol{K}$ collects all pairwise correlations. Usually, correlation functions should be positive semi-definite (PSD), i.e., all eigenvalues of $\boldsymbol{K}$ are non-negative. The Kriging predictor is:

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{k}^T \boldsymbol{K}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}),$$

where $\mathbf{y}$ are the training observations, $\hat{\mu}$ represents the process mean, $\mathbf{1}$ is a vector of ones and $\boldsymbol{k}$ is the column vector of correlations between the set of training samples $\boldsymbol{X}$ and the new sample $\boldsymbol{x}$. All parameters are usually determined by Maximum Likelihood Estimation (MLE). Kriging is a popular choice in SMBO algorithms, as it provides an estimate of the prediction uncertainty:

$$\hat{s}^2(\boldsymbol{x}) = \hat{\sigma}^2(1 - \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{k}),$$

where the process variance $\hat{\sigma}$ is also determined by MLE. The estimate $\hat{s}^2(\boldsymbol{x})$ can be used to balance exploration and exploitation by computing the Expected Improvement (EI) of candidate solutions [13]. The EI is a frequently employed infill criterion, e.g., in EGO [9].

Kriging also allows to deal with noisy data, using the so called nugget effect. The nugget adds a small constant $\eta > 0$ to the diagonal of $\boldsymbol{K}$. Thus, the otherwise interpolating Kriging model is able to regress the data, introducing additional smoothness into the predicted value. The nugget effect may also help to increase the numerical stability. A re-interpolation approach can be used to avoid that the nugget effect deteriorates the uncertainty estimate [12].

4 Kernels for Hierarchical Search Spaces

Hierarchical variables can be defined as variables that are only *active* if other variables fulfill a condition. An *active* variable has an impact on the objective function value. We use the notation of Hutter and Osborne [4]: a function $\delta_i(\boldsymbol{x})$ determines whether the i-th variable of $\boldsymbol{x}$ is active ($\delta_i(\boldsymbol{x}) = \text{true}$) or not (false). In the following, only the per-variable distance $\mathrm{d}_i(x_i, x_i')$ will be introduced for each kernel. The combined kernel structure is identical for all cases unless stated otherwise, i.e., $\mathrm{k}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\sum_{i=1}^{d} \mathrm{d}_i(x_i, x_i'))$. We describe an existing kernel (Arc) and propose four alternatives (Ico, IcoCorrected, Imp, ImpArc).

4.1 The Arc-kernel

The Arc-kernel proposed by Hutter and Osborne [4] is specifically developed to handle hierarchical structures. It is based on three assumptions. First, if a hierarchical variable is inactive in two configurations $\boldsymbol{x}$ and $\boldsymbol{x}'$, then the distance in that dimension should be zero. Second, if it is active in both configurations, the distance depends on the respective variable values. Third, if the variable is

only active in one configuration, the distance should be a constant, because no information is available to compare an inactive with an active variable.

An embedding is required to encode these assumptions in valid distance measures that yield a PSD kernel. It is for continuous variables [4]:

$$\mathrm{d}_{\mathrm{Arc}i}(x_i, x_i') = \begin{cases} 0, & \text{if } \delta_i(\boldsymbol{x}) = \delta_i(\boldsymbol{x}') = \text{false} \\ \theta_i, & \text{if } \delta_i(\boldsymbol{x}) \neq \delta_i(\boldsymbol{x}') \\ \theta_i \sqrt{2 - 2\cos(\pi \rho_i \frac{x_i - x_i'}{u_i - l_i})}, & \text{if } \delta_i(\boldsymbol{x}) = \delta_i(\boldsymbol{x}') = \text{true} \end{cases} \tag{1}$$

The kernel variables $\theta_i \in \mathbb{R}^+$ and $\rho_i \in [0, 1]$ are determined by MLE. A respective measure for categorical variables can be found in [4]. We follow up on [5] and skip the notion of putting further restrictions on θ_i to encode lower importance of lower hierarchical levels as proposed in [4]. Moreover, we use the square of the distance in the embedded space (i.e., removing the square root in Eq. (1)), since we also use squared deviations in all other distances.

4.2 Indefinite Conditional Kernel

We propose a simplified alternative to the Arc-kernel:

$$\mathrm{d}_{\mathrm{Ico}i}(x_i, x_i') = \begin{cases} 0, & \text{if } \delta_i(\boldsymbol{x}) = \delta_i(\boldsymbol{x}') = \text{false} \\ \rho_i, & \text{if } \delta_i(\boldsymbol{x}) \neq \delta_i(\boldsymbol{x}') \\ \theta_i \mathrm{d}_i(x_i, x_i'), & \text{if } \delta_i(\boldsymbol{x}) = \delta_i(\boldsymbol{x}') = \text{true} \end{cases}$$

Here, $\mathrm{d}_i(x_i, x_i')$ is an appropriate default distance (numerical: square deviation $(x_i - x_i')^2$, categorical: Hamming distance). The distance parameter $\rho_i \in \mathbb{R}^+$ is determined by MLE. The kernel follows the same intuitive assumptions as $\mathrm{d}_{\mathrm{Arc}}$, but it does not use the complicated cylindrical embedding. This may lead to indefinite kernel matrices for some data sets or choices of parameters. Due to this, it will be denoted as the indefinite conditional kernel, or Ico-kernel.

As a variant of the Ico-kernel, the IcoCorrected (IcoCor) kernel is the same kernel subject to a correction via a spectrum-flip. This transformation of the eigenspectrum generates PSD kernel matrices from indefinite kernels, cf. [14]. Note, that the nugget effect may also correct issues with definiteness if η is large enough. Thus, even the uncorrected Ico-kernel can produce a valid model.

4.3 Imputation Kernel

Alternatively, we propose a simple PSD kernel. It is based on a different assumption: If the hierarchical variable is only active in one of two configurations ($\delta_i(\boldsymbol{x}) \neq \delta_i(\boldsymbol{x}')$), their distance in that dimension is *not* assumed to be constant. Rather, it is assumed that the value of the active configuration does influence the dissimilarity. This is achieved by introducing a kernel parameter against which

the respective active value is compared. Thus,

$$\mathrm{d}_{\mathrm{Imp}_i}(x_i, x_i') = \begin{cases} 0, & \text{if } \delta_i(\boldsymbol{x}) = \delta_i(\boldsymbol{x}') = \text{false} \\ \theta_i \mathrm{d}_i(x_i', \rho_i), & \text{if } \delta_i(\boldsymbol{x}) = \text{false} \neq \delta_i(\boldsymbol{x}') \\ \theta_i \mathrm{d}_i(x_i, \rho_i), & \text{if } \delta_i(\boldsymbol{x}) = \text{true} \neq \delta_i(\boldsymbol{x}') \\ \theta_i \mathrm{d}_i(x_i, x_i'), & \text{if } \delta_i(\boldsymbol{x}) = \delta_i(\boldsymbol{x}') = \text{true} \end{cases}$$

where d_i is again the appropriate default distance (square deviation, Hamming) and ρ_i is of the same data type as x_i. For real x_i, the bounds of x_i and ρ_i can differ. We use $\rho_i \in [l_i - a, u_i + a] \subset \mathbb{R}$ with $a = 2 * (u_i - l_i)$. Larger bounds may be necessary, depending on the problem. Similarly, if x_i is categorical ρ_i can have one more level (category) than x_i, to emulate the case where none of the other levels is a good replacement. An exponential kernel based on $\mathrm{d}_{\mathrm{Imp}_i}(x_i, x_i')$ can be proven to be PSD. Using proposition 2 in [4], we only need to show that there exists a mapping function $f_i(x_i)$ that maps to a space in which a valid distance can be used, i.e., $\mathrm{d}_{\mathrm{Imp}_i}(x_i, x_i') = \mathrm{d}_i(f_i(x_i), f_i(x_i'))$. For $\mathrm{d}_{\mathrm{Imp}}$, the mapping function is

$$f_i(x_i) = \begin{cases} x_i \text{ if } \delta_i(\boldsymbol{x}) \\ \rho_i \text{ otherwise.} \end{cases}$$

Hence, the resulting kernel based on $\mathrm{d}_i(f_i(x_i), f_i(x_i'))$ is PSD.

Clearly, this kernel has relations to the imputation approach mentioned in Sec. 1. Essentially, inactive values are replaced by an imputed value ρ_i. Instead of choosing that value a-priori, it is defined as a parameter and determined by MLE. Hence, it will be denoted as the imputation kernel or Imp-kernel. One drawback of this kernel is, that if x_i is categorical, ρ_i is also categorical. This may complicate the MLE procedure. Also, the assumption that some value can be imputed is less conservative than the assumptions of the Arc-kernel.

4.4 The Imputation-Arc Kernel

When it is unclear whether the Arc- or Imp-kernel is more appropriate, we suggest a linear combination denoted as the ImpArc-kernel,

$$\mathrm{k}_{\mathrm{ImpArc}}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\sum_{i=1}^{d} \beta_{1,i} \mathrm{d}_{\mathrm{Arc}\,i}(x_i, x_i') + \beta_{2,i} \mathrm{d}_{\mathrm{Imp}_i}(x_i, x_i')),$$

with weights $\beta_{k,i} \in \mathbb{R}^+$ determined by MLE. Other combinations (e.g., Ico-Imp, Imp-Arc-Ico) are possible. We only test the ImpArc combination, because the Ico- and Arc-kernel express very similar information. Also, a three-way combination would require to learn an additional weight $\beta_{3,i}$.

5 Experimental Setup

While synthetic, tree-based test functions for hierarchical search spaces have been proposed by Jenatton et al. [8], they are not able to respect the different

definitions and assumptions of our kernels. Hence, we suggest a simple two-dimensional quadratic function

$$f(\boldsymbol{x}) = (x_1 - d)^2 + \begin{cases} 0 & \text{if} \quad x_1 \leq c \\ (x_2 - 0.5)^2 + b & \text{else} \end{cases}.$$

The function's behavior (see Fig. 1) is defined by the constants b, c and d. The constant b controls whether the Imp-kernel is a good match, c controls the size of the active region and d controls the location of the optimum. The function is influenced by the hierarchical variable x_2 only if $x_1 > c$ and does have a discontinuity at $x_1 = c$. For $b = 0$, the function is continuous at $x_2 = 0.5$. Hence, the if-else term of $f(\boldsymbol{x})$ yields identical results if $x_2 = 0.5$ and if $\delta_2(\boldsymbol{x}) = false$. In this case, the assumption of the Imp-kernel is fulfilled, i.e., the kernel definition matches the problem structure. The Imp-kernel should learn to impute $\rho = 0.5$.

We identified five situations with different expected performances.

A) $d < c$ (the optimum is in the *inactive* region of x_2 at $x_1 = d, x_2 \in \mathbb{R}$) and $b = 0$ (imputation potentially *profitable*). The function is *unimodal*.
B) $d < c$ (the optimum is in the *inactive* region at $x_1 = d, x_2 \in \mathbb{R}$) but $b > 0$ (imputation potentially *unprofitable*). The function is *unimodal*.
C) $d > c$ (the optimum is in the *active* region at $x_1 = d, x_2 = 0.5$) and $b = 0$ (imputation potentially *profitable*). The function is *bimodal*.
D) $d > c$ (the optimum is in the *active* region at $x_1 = d, x_2 = 0.5$) and $b = 0.1$ (imputation potentially *unprofitable*) and $b < (c - d)^2$. The function is *bimodal*. The discontinuity at c is not as important, since the optimum is remote from it.
E) $d > c$ (the optimum is in the *active* region at $x_1 = c, x_2 \in \mathbb{R}$) and $b = 0.1$ (imputation potentially *unprofitable*) and $b > (c - d)^2$. The function is *bimodal*. The discontinuity at c has to be approximated well, since the optimum is at $x_1 = c$.

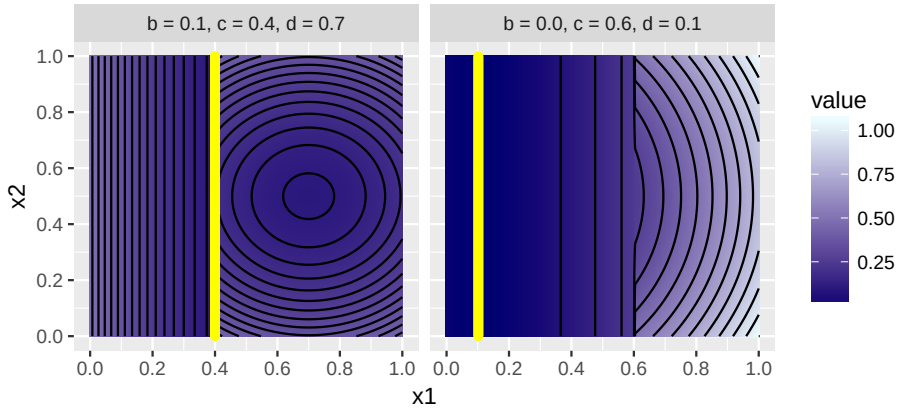

Fig. 1. Visualization of the test function, the optimum is marked in yellow.

Covering all of these five situations, we tested all combinations of the values $b = \{0, 0.1\}$, $c = \{0.2, 0.4, 0.6, 0.8\}$, and $d = \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

To estimate model quality, we measured the model's Root Mean Squared Error (RMSE). The models were trained with 10, the error was estimated on 1 000 uniform random samples. The Kriging model was trained with the `CEGO` package in R [15, 16]. It was configured to use the nugget effect and re-interpolation. The Dividing Rectangles algorithm [17] was chosen to optimize the model parameters via 200 likelihood evaluations. We used all kernels from Sec. 4 and a standard exponential kernel with square deviation in each dimension (which does not incorporate hierarchical information), denoted as the Stan-kernel.

The same type of model was used in the SMBO algorithm from the `CEGO` package. The search was limited to 10 evaluations of $f(\boldsymbol{x})$, due its low difficulty, low dimensionality and assumed cost. The search was initialized with three uniform random samples. Based on the model, the EI criterion was optimized by DE [11]. We used the `DEoptim` package [18] with 10 000 EI evaluations per iteration and used default parameters otherwise. Each experiment was repeated 100 times, with 100 unique random seeds (one per replication). We recorded the difference between the best found and the optimal function value (*suboptimality*) for each replication.

6 Results

First, we analyze the model quality produced by the different kernels. Fig. 2 shows the median RMSE value for all parameter constellations and kernels. Clearly, the fit of the Stan-kernel is inferior to most specialized hierarchical kernels for almost all parameter constellations, especially if $b = 0.1$.

If $b = 0$, the assumption of the Imp-kernel is fulfilled. Hence, both the Imp- and the ImpArc-kernel produce a better fit than most other kernels. However, for $b = 0.1$, the Imp-kernel mostly has the second or third worst performance. Only the Stan-kernel and sometimes the IcoCorrected-kernel perform worse. The Arc- and the Ico-kernel achieve very similar performances in most cases, with near-to-best performance if $b = 0.1$. The ImpArc-kernel, combining the advantages of the Arc- and the Imp-kernel, has a good, sometimes best fit in all situation, for both $b \in \{0, 0.1\}$. Contrarily, the IcoCorrected-kernel has a rather poor fit in several cases, sometimes even worse than the Stan-kernel. Overall, differences between kernels tend to disappear for large values of c, which is to be expected due to the reduced influence of the hierarchical variable x_2.

To get a better understanding of the kernels, we visualize an example for Situation E with $(b, c, d) = (0.1, 0.4, 0.7)$. Fig. 3 shows line plots for the test function as well as fitted models for all six kernels, trained with ten uniform random samples. Here, the global optimum is at $x_1 = c = 0.4$, i.e., at the jump discontinuity. The function value of the global optimum (0.09) is only slightly better than the value of the local optimum (0.1) at $(x_1 = 0.7, x_2 = 0.5)$. Hence, to find the global optimum, it is important to model the discontinuity well.

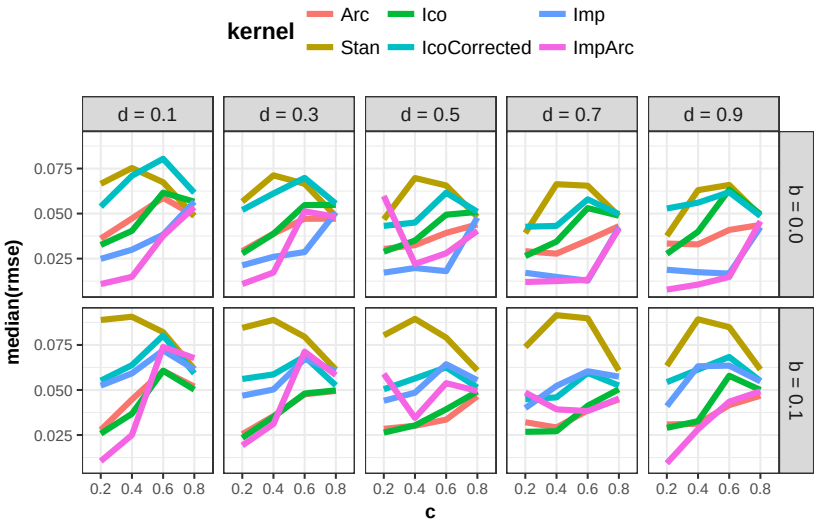

Fig. 2. Median RMSE values over 100 replications for all configurations and kernels.

The Stan-kernel is not able to model the discontinuity and therefore tries to fit a smooth curve to the function. Hence, the Stan-kernel approximates the optimum poorly. For $x_1 = 0.5$ the model of the Imp-kernel shares the poor performance of the Stan kernel: It is not able to fit the discontinuity. Still, the fit is much closer to the true objective function. For $x_1 = 0.75$ the Imp-kernel is able to fit the discontinuity, but the fit is inferior to the Arc-, Ico- and ImpArc-kernel. All of them reproduce the discontinuity quite well. However, their approximation of the function for $x_1 > c, x_2 = 0.75$ has a strong offset. While this is not a perfect fit, it will not necessarily deteriorate optimization performance. The model based on the IcoCorrected-kernel is able to reproduce the discontinuity, but the jump is not large enough to identify the optimum at $x_1 = 0.4$.

Next, we analyze the optimization performance. Due to space restrictions, we present statistical test results that summarize the experimental data. Following Demšar [19], we apply Friedman and corresponding post-hoc Nemenyi tests in order to find significant differences between the kernels, using the function parameters b, c and d as blocking variables for the tests. We extend Demšar's approach, since we do not apply our tests to the median suboptimalities. Instead, we use the replication identifier as an additional blocking variable. This accounts for the effect of the initial design. We visualize the test results using ordered graphs that present a rough order on the kernels.

We start by investigating the combined results of all optimization experiments. With a p-value that is numerically approximating zero ($< 10^{-16}$), the Friedman-test indicates that there are significant differences between the different kernels. Note, if differences are present p-values tend to be small due to the

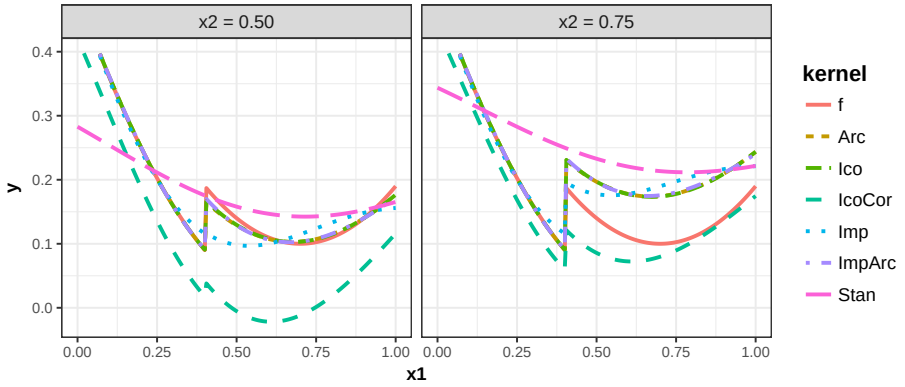

Fig. 3. Example fits of the kernels. Two slice planes are shown for $x_2 \in \{0.5, 0.75\}$.

large number of experiment replications, and differentiating between significant and relevant differences is an open issue in the analyses of computer experiments.

Fig. 4 shows the results of the corresponding Nemenyi-test, including a graph representation of the test results as well as mean ranks for each kernel. As expected, the Stan-kernel is clearly outperformed by all other kernels. For the other kernels, we can identify two groups: The Imp- and the ImpArc-kernel seem to perform slightly better than the rest. Within each group, there are no significant differences between the kernels, while tests between kernel from groups are significant. It is questionable how reliable this result is. We expect diverse behavior of the kernels in the five situation and the overall performance is of course influenced by the selection of the specific test instances. Hence, we will now examine individual tests for situations A to E.

As in the global situation, all Friedman-tests result into very small p-values (numerically approximating zero). Hence, there is evidence for significant differences between at least some kernels in each situation. Fig. 5 shows the results

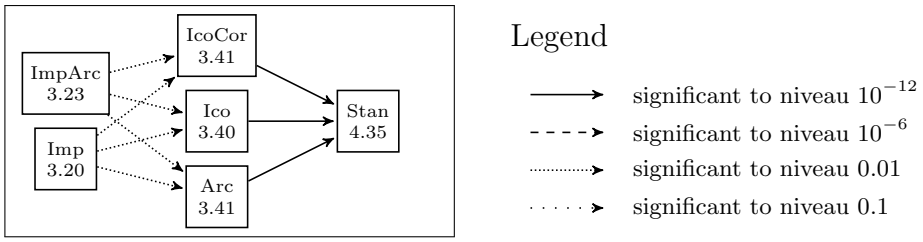

Fig. 4. Ordering of the six kernels with respect to their mean ranks (printed below each kernel) over all test instances. A path (possibly using multiple edges) between two kernels denotes a significant difference of the post-hoc Nemenyi test, the directions of the arrows follows the ordering of the mean ranks.

of the post-hoc Nemenyi-tests in all five situations. In situation A and C, the assumption of the Imp-kernel is fulfilled, since $b = 0$ allows for imputation. This is reflected by the results: In both situations A and C the Imp-kernel performs best. The Imp-kernel outperforms the Arc- and Ico-kernel with a large margin in situation C. Contrarily, in situation B (unimodal, not imputable) and E (bimodal, not imputable), where the imputation assumption is violated, the Imp-kernel performance is inferior. These observations fit to our expectations: b controls whether or not the Imp-kernel is able to find a good value to impute.

The Arc- and the Ico-kernel have similar results in most situations, except for situation C. This confirms that these kernels encode similar information, and it also shows that the indefiniteness of the Ico-kernel does not seem to impact optimization performance. At least, the indefiniteness is sufficiently well mitigated by the employed nugget effect.

While performing reasonably well, the ImpArc-kernel never achieves a top performance. It is usually positioned in the second-best group. This can be explained by the fact that it attains some middle ground between the kernels that

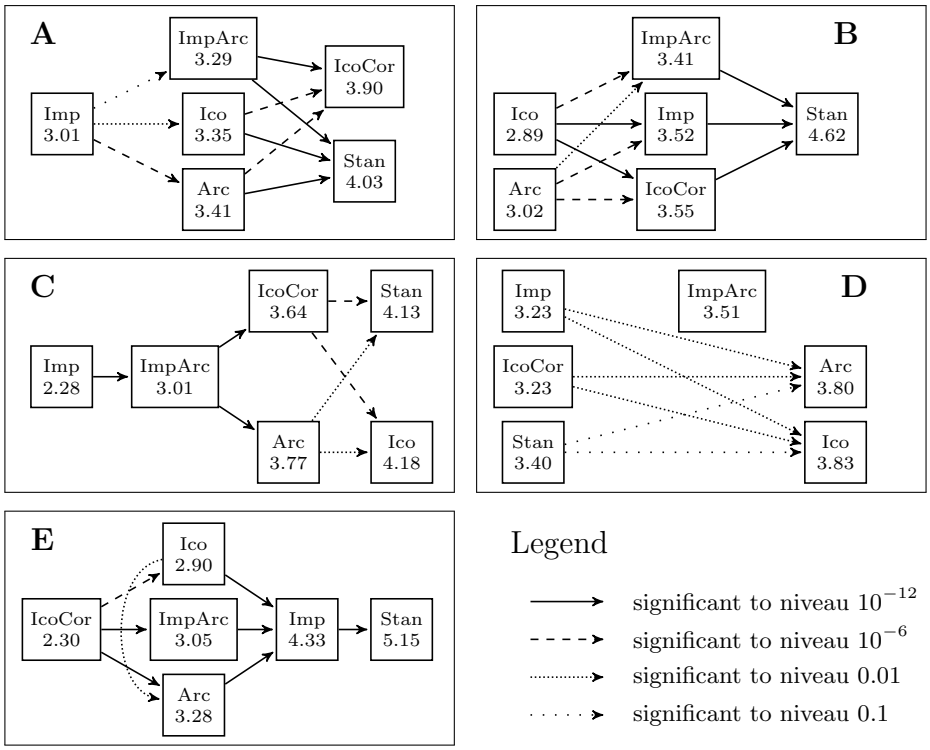

Fig. 5. Ordering of the kernels in the five situations with respect to their mean ranks (printed below each kernel) over all test instances. A path (possibly using multiple edges) between two kernels denotes a significant difference of the post-hoc Nemenyi test the directions of the arrows follows the ordering of the mean ranks.

it combines. The IcoCorrected-kernel performs poorly in some situations (A, B, C), but it performs best in situation E. Poor performance may be caused by inconsistencies in the employed definiteness repair methods. However, it remains unclear to us why the performance is distinctively better in situation E.

Situation D (bimodal, not imputable) has a rather special behavior. Only few distinct differences between the kernels can be detected. Moreover, it is the only situation in which the Stan kernel does not perform in the worst group. We suggest that this is due to the fact that modeling the discontinuity is not as important here. The optimum lies in the region where x_2 is active, hence it may even be detrimental to model the discontinuity. That means, if the optimum is far enough from the discontinuity, it may be helpful to smoothen through the local optimum that lies at the discontinuity, since this will drive the search towards the global optimum. This could also explain why the Imp-kernel outperforms the Arc-kernel in situation D, despite $b = 0.1$.

7 Conclusion and Outlook

We investigated different kernels for SMBO in hierarchical search spaces, e.g., the Arc-kernel previously proposed by Hutter and Osborne [4], the Ico-kernel which is similar, yet indefinite, and the Imp-kernel which attempts to learn suitable imputed values for inactive variables. We tested both the model quality and the optimization performance of six kernels, and received consistent results. Hence, we can answer our research questions and deduct simple recommendations for choosing a kernel.

1. The hierarchical structure should be incorporated into the kernel.
2. The Imp-kernel should be chosen if it is a-priori known that its assumption is fulfilled. If the assumption is violated, the Arc- and Ico-kernel are good choices. Without prior knowledge, the ImpArc-kernel is a sound compromise.
3. We did not observe many significant differences between the Arc- and the Ico-kernel. The kernels' definiteness does not seem to have a strong impact.

These result rely on tests with a rather simple test function, and hence have to interpreted with care. Devising more complex test functions with higher input dimensions is clearly of interest. But while artificial tests are instructive due to their controlled behavior, it is not always clear how this translates to real world problems. Hence, it would be desirable to make tests with real world applications, such as algorithm tuning.

Furthermore, it would be interesting to let the infill optimizer exploit the information on variable activity, to avoid searching in inactive areas of the search space. The same is true for the initialization of the SMBO algorithm. Spreading a space-filling design in inactive areas is wasteful.

Finally, all discussed distances $\mathrm{d}_i(x_i, x_i')$ are defined for a single dimension i. Therefore, we are not limited to a single choice. Rather, different distances can be chosen for each dimension (e.g., Arc for x_i, and Imp for $x_{j \neq i}$).

References

1. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA. In: Proc. of the 19th Int. Conf. on Knowledge Discovery and Data Mining, ACM Press (2013)
2. Horn, D., Bischl, B.: Multi-objective parameter configuration of machine learning algorithms using model-based optimization. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI). (2016)
3. Cáceres, L.P., Bischl, B., Stützle, T.: Evaluating random forest models for irace. In: Proc. of the Genetic and Evolutionary Computation Conf., ACM Press (2017)
4. Hutter, F., Osborne, M.A.: A kernel for hierarchical parameter spaces. Technical Report arXiv:1310.5738, arXiv (2013)
5. Swersky, K., Duvenaud, D., Snoek, J., Hutter, F., Osborne, M.: Raiders of the lost architecture: Kernels for bayesian optimization in conditional parameter spaces. In: NIPS workshop on Bayesian Optimization in Theory and Practice. (2013)
6. Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: Advances in Neural Information Processing Systems 24. Curran Associates, Inc. (2011)
7. Bergstra, J., Yamins, D., Cox, D.: Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In: Proc. of the 30th Int. Conf. on Machine Learning, PMLR (2013)
8. Jenatton, R., Archambeau, C., González, J., Seeger, M.: Bayesian optimization with tree-structured dependencies. In: Proc. 34th Int. Conf. on Machine Learning, PMLR (2017)
9. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. Journal of Global Optimization **13**(4) (1998)
10. Bischl, B., Richter, J., Bossek, J., Horn, D., Thomas, J., Lang, M.: mlrmbo: A modular framework for model-based optimization of expensive black-box functions. arXiv preprint arXiv:1703.03373 (2017)
11. Storn, R., Price, K.: Differential evolution — a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization **11**(4) (1997)
12. Forrester, A., Sobester, A., Keane, A.: Engineering Design via Surrogate Modelling. Wiley (2008)
13. Mockus, J., Tiesis, V., Zilinskas, A.: The application of Bayesian methods for seeking the extremum. In: Towards Global Optimization 2. North-Holland (1978)
14. Zaefferer, M., Bartz-Beielstein, T.: Efficient global optimization with indefinite kernels. In: Parallel Problem Solving from Nature–PPSN XIV, Springer (2016)
15. Zaefferer, M.: Combinatorial efficient global optimization in R - CEGO v2.2.0. online: https://cran.r-project.org/package=CEGO (2017) accessed: 2018-01-10.
16. Zaefferer, M., Stork, J., Friese, M., Fischbach, A., Naujoks, B., Bartz-Beielstein, T.: Efficient global optimization for combinatorial problems. In: Proc. of the Genetic and Evolutionary Computation Conf., ACM (2014)
17. Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitzian optimization without the lipschitz constant. Journal of Optimization Theory and Applications **79**(1) (1993)
18. Mullen, K., Ardia, D., Gil, D., Windover, D., Cline, J.: DEoptim: An R package for global optimization by differential evolution. Journal of Statistical Software **40**(6) (2011)
19. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. **7** (2006)