Expected Improvement versus Predicted Value in Surrogate-Based Optimization

Frederik Rehbach, Martin Zaefferer, Boris Naujoks, Thomas Bartz-Beielstein
Institute for Data Science, Engineering, and Analytics
TH Köln, Gummersbach, Germany
firstname.surname@th-koeln.de

ABSTRACT

Surrogate-based optimization relies on so-called infill criteria (acquisition functions) to decide which point to evaluate next. When Kriging is used as the surrogate model of choice (also called Bayesian optimization), one of the most frequently chosen criteria is expected improvement. We argue that the popularity of expected improvement largely relies on its theoretical properties rather than empirically validated performance. Few results from the literature show evidence, that under certain conditions, expected improvement may perform worse than something as simple as the predicted value of the surrogate model. We benchmark both infill criteria in an extensive empirical study on the 'BBOB' function set. This investigation includes a detailed study of the impact of problem dimensionality on algorithm performance. The results support the hypothesis that exploration loses importance with increasing problem dimensionality. A statistical analysis reveals that the purely exploitative search with the predicted value criterion performs better on most problems of five or higher dimensions. Possible reasons for these results are discussed. In addition, we give an in-depth guide for choosing the infill criteria based on prior knowledge about the problem at hand, its dimensionality, and the available budget.

CCS CONCEPTS

• **Theory of computation → Mathematical optimization**; **Gaussian processes**; • **Computing methodologies → Modeling and simulation**;

KEYWORDS

Surrogate-based Optimization, Bayesian Optimization, Infill Criterion, Acquisition Function

ACM Reference Format:
Frederik Rehbach, Martin Zaefferer, Boris Naujoks, Thomas Bartz-Beielstein. 2020. Expected Improvement versus Predicted Value in 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/10.1145/3377930.3389816

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
GECCO '20, July 8–12, 2020, Cancún, Mexico
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7128-5/20/07…$15.00
https://doi.org/10.1145/3377930.3389816

1 INTRODUCTION

Many real-world optimization problems require significant resources for each evaluation of a given candidate solution. For example, evaluations might require material costs for laboratory experiments or computation time for extensive simulations. In such scenarios, the available budget of objective function evaluations is often severely limited to a few tens or hundreds of evaluations.

One standard method to efficiently cope with such limited evaluation budgets is surrogate-based optimization (SBO). Surrogate models of the objective function are based on data-driven models, which are trained using only a relatively small set of observations. Predictions from these models partially replace expensive objective function evaluations. Since the surrogate model is much cheaper to evaluate than the original objective function, an extensive search becomes feasible. In each SBO iteration, one new candidate solution is proposed by the search on the surrogate. The candidate is evaluated on the expensive function and the surrogate is updated with the new data. This process is iterated until the budget of expensive evaluations is depleted.

A frequently used surrogate model is Kriging (also called Gaussian process regression). In Kriging, a distance-based correlation structure is determined with the observed data [12]. The search for the next candidate solution on the Kriging model is guided by a so-called infill criterion or acquisition function.

A straightforward and simple infill criterion is the predicted value (PV). The PV of a Kriging model is an estimation or approximation of the objective function at a given point in the search space [12]. If the surrogate exactly reproduces the expensive objective function, then optimizing the PV yields the global optimum of the expensive objective function. In practice, especially in the early iterations of the SBO procedure, the prediction of the surrogate will be inaccurate. Many infill criteria do not only consider the raw predicted function value but also try to improve the model quality with each new iteration. This is achieved by suggesting new solutions in promising but unknown regions of the search space. In essence, such criteria attempt to simultaneously improve the local approximation quality of the optimum as well as the global prediction quality of the surrogate.

The expected improvement (EI) criterion is considered as a standard method for this purpose [20]. EI makes use of the internal uncertainty estimate provided by Kriging. The EI of a candidate solution increases if the predicted value or the estimated uncertainty of the model rises.

The optimization algorithm might converge to local optima, if solely the PV is used as an infill criterion. In the most extreme case, if the PV suggests a solution that is equal to an already known solution, the algorithm might not make any progress at all, instead

repeatedly suggesting the exact same solution [12]. Conversely, EI can yield a guarantee for global convergence, and convergence rates can be analyzed analytically [8, 33]. As a result, EI is frequently used as an infill criterion.

We briefly examined 24 software frameworks for SBO. We found 15 that use EI as a default infill criterion, three that use PV, five that use criteria based on lower or upper confidence bounds [1], and a single one that uses a portfolio strategy (which includes confidence bounds and EI, but not PV). An overview of the surveyed frameworks is included in the supplementary material. Seemingly, EI is firmly established.

Despite this, some criticism can be found in the literature. Wessing and Preuss point out that the convergence proofs are of theoretical interest but have no relevance in practice [33]. The most notable issue in this context is the limited evaluation budget that is often imposed on SBO algorithms. Under these limited budgets, the asymptotic behavior of the algorithm can become meaningless. Wessing and Preuss show results where a model-free CMA-ES often outperforms EI-based EGO within a few hundred function evaluations [33].

A closely related issue is that many infill criteria, including EI, define a static trade-off between exploration and exploitation [13, 21, 30, 31]. This may not be optimal. Intuitively, explorative behavior may be of more interest in the beginning, rather than close to the end of an optimization run.

At the same time, benchmarks that compare EI and PV are far and few between. While some benchmarks investigate the performance of EI-based algorithms, they rarely compare to a variant based on PV (e.g., [17, 23, 25, 27]). Other authors suggest portfolio strategies that combine multiple criteria. Often, tests of these strategies do not include the PV in the portfolio (e.g., [10, 18]).

We would like to discuss two exceptions. Firstly, Noe and Husmeier [24] do not only suggest a new infill criterion and compare it to EI, but they also compare it to a broad set of other criteria, importantly including PV. In several of their experiments, PV seems to perform better than EI, depending on the test function and the number of evaluations spent. Secondly, a more recent work by De Ath et al. [9] reports similar observations. They observe that"*Interestingly, Exploit, which always samples from the best mean surrogate prediction is competitive for most of the high dimensional problems*" [9]. Conversely, they observe better performances of explorative approaches on a few, low-dimensional problems.

In both studies, the experiments cover a restricted set of problems where the dimensionality of each specific test function is usually fixed. The influence of test scenarios, especially in terms of the search space dimension, remains to be clarified. It would be of interest, to see how increasing or decreasing the dimension affects results for each problem.

Hence, we raise two tightly connected research questions:

RQ-1 Can distinct scenarios be identified where PV outperforms EI or vice versa?

RQ-2 Is EI a reasonable choice as a default infill criterion?

Here, scenarios comprehend the whole search framework, such as features of the problem landscape, dimension of the optimization problem, or the allowed evaluation budget. In the following, we describe experiments that investigate these research questions. We discuss the results based on our observations and a statistical analysis.

2 EXPERIMENTS

2.1 Performance Measurement

Judging the behavior of an algorithm requires an appropriate performance measure. The choice of this measure depends on the context of our investigation. A necessary pretext for our experiments is the limited budget of objective evaluations we allow for focusing on algorithms for expensive optimization problems. Since the objective function causes the majority of the optimization cost in such a scenario, a typical goal is to find the best possible candidate solution within a given amount of evaluations.

For practical reasons, we have to specify an upper limit of function evaluations for our experiments. However, for our analysis, we evaluate the performance measurements for each iteration. Readers who are only interested in a scenario with a specific fixed budget can, therefore, just consider the results of the respective iteration.

2.2 Algorithm Setup

One critical contribution to an algorithm's performance is its configuration and setup. This includes parameters as well as the chosen implementation.

All configurations and codes that are discussed in the following are available on GitHub[1]. We chose the R-package 'SPOT' [2, 3] as an implementation for SBO. It provides various types of surrogate models, infill criteria, and optimizers. To keep the comparison between the EI and PV infill criteria as fair as possible, they will be used with the same configuration in 'SPOT'. This mentioned configuration regards the optimizer for the model, the corresponding budgets, and lastly the kernel parameters for the Kriging surrogate. The configuration is explained in more detail in the following.

We allow each 'SPOT' run to expend 300 evaluations of the objective function. This accounts for the assumption that SBO is usually applied in scenarios with severely limited evaluation budgets. This specification is also made due to practical reasons: the runtime of training Kriging models increases exponentially with the number of data points. Hence, a significantly larger budget would limit the number of tests we could perform due to excessive algorithm runtimes. Larger budgets, for instance, 500 or 1 000 evaluations, may easily lead to infeasible high runtimes (depending on software implementation, hardware, and the number of variables). If necessary, this can be partially alleviated with strategies that reduce runtime complexity, such as clustered Kriging [29, 32]. However, these strategies have their own specific impact on algorithm performance, which we do not want to cover in this work.

The first ten of the 300 function evaluations are spent on an initial set of candidate solutions. This set is created with Latin Hypercube sampling [22]. In the next 290 evaluations, a Kriging surrogate model is trained at each iteration. The model is trained with the 'buildKriging' method from the 'SPOT' package. This Kriging implementation is based on work by Forrester et. al. [12],

[1] https://github.com/frehbach/rehb20a

Table 1: Overview of the 24 BBOB test functions including some of their landscape features.

ID	Name	Specific Features
1	Sphere	unimodal, separable, symmetric
2	Ellipsoidal	unimodal, separable, high conditioning
3	Rastrigin	multimodal, separable, regular/symmetric structure
4	Büche-Rastrigin	multimodal, separable, asymmetric structure
5	Linear Slope	unimodal, separable
6	Attractive Sector	unimodal, low/moderate conditioning , asymmetric structure
7	Step Ellipsoidal	unimodal, low/moderate conditioning, many plateaus
8	Rosenbrock	unimodal/bimodal depending on dimension, low/moderate conditioning
9	Rosenbrock, rotated	unimodal/bimodal depending on dimension, low/moderate conditioning
10	Ellipsoidal	unimodal, high conditioning
11	Discus	unimodal, high conditioning
12	Bent Cigar	unimodal, high conditioning
13	Sharp Ridge	unimodal, high conditioning
14	Different Powers	unimodal, high conditioning
15	Rastrigin	multimodal, non-separable, low conditioning, adequate global structure
16	Weierstrass	multimodal (with several global otpima), repetitive, adequate global structure
17	Schaffers F7	multimodal, low conditioning, adequate global structure
18	Schaffers F7	multimodal, moderate conditioning, adequate global structure
19	Griwank-Rosenbrock	multimodal, adequate global structure
20	Schwefel	multimodal, weak global structure (in the optimal, unpenalized region)
21	Gallagher G.101-me Peaks	multimodal, low conditioning, weak global structure
22	Gallagher G. 21-hi Peaks	multimodal, moderate conditioning, weak global structure
23	Katsuura	multimodal, weak global structure
24	Lunacek bi-Rastrigin	multimodal, weak global structure

and uses an anisotropic kernel:

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\sum_{i=1}^{d} -\theta_i |x_i - x_i'|^{p_i}\right).$$

Here, $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$ are candidate solution vectors, with dimension d and $i = 1, \ldots, d$. Furthermore, the kernel has parameters $\theta_i \in \mathbb{R}_+$ and $p_i \in [0.01, 2]$. During the model building, the nugget effect [12] is activated for numerical stability, and the respective parameter λ is set to a maximum of 10^{-4}. Parameters θ_i, p_i and λ are determined by Maximum Likelihood Estimation (MLE) via differential evolution [28] ('optimDE'). The budget for the MLE is set to $500 \times t$ likelihood evaluations, where $t = 2d + 1$ is the number of model parameters optimized by MLE (θ,p, λ).

The respective infill criterion is optimized, after the model training. Again, differential evolution is used for this purpose. In this case, we use a budget of $1\,000 \times d$ evaluations of the model (in each iteration of the SBO algorithm). Generally, these settings were chosen rather generously to ensure high model quality and a well-optimized infill criterion.

To rule out side effects caused by potential peculiarities of the implementation, independent tests were performed with a different implementation based on the R-package 'mlrmbo' [4, 5]. Notably, the tests with 'mlrmbo' employed a different optimizer (focus search instead of differential evolution) and a different Kriging implementation, based on DiceKriging [26].

2.3 Test Functions

To answer the presented research questions, we want to investigate how different infill criteria for SBO behave on a broad set of test functions. We chose one of the most well-known benchmark suites in the evolutionary computation community: the Black-Box Optimization Benchmarking (BBOB)[2] suite [14]. While the test functions in the BBOB suite themselves are not expensive to evaluate, we emulate expensive optimization by limiting the algorithms to only a few hundred evaluations. The standard BBOB suite contains 24 noiseless single-objective test functions, divided into five groups by common features and landscape properties. The global optima and landscape features of all functions are known. An overview of the 24 functions and their most important properties is given in Table 1. Hansen et al. present a detailed description [16].

Each of the BBOB test functions is scalable in dimensionality so that algorithms can be compared with respect to performance per dimension. Furthermore, each function provides multiple instances. An instance is a rotated or shifted version of the original objective function. All described experiments were run with a recent GitHub version[3], v2.3.1 [15]. Each algorithm is tested on the 15 available standard instances of the BBOB function set.

Preliminary experiments were run using the smoof test function set implemented in the 'smoof' R-package [6, 7]. The smoof package consists of a total of more than 70 single-objective functions as well as 24 multi-objective functions. Since our analysis will consider observations in relation to the scalable test function

[2] https://coco.gforge.inria.fr/
[3] https://github.com/numbbo/coco

dimension, we do not further discuss the complete results of the (often non-scalable) smoof test function set. Yet, considering the fixed dimensions in smoof, similar observations were made for both smoof and BBOB.

3 RESULTS

3.1 Convergence Plots

Our first set of results is presented in the form of convergence plots shown in Figure 1. The figure shows the convergence of the SBO-PV and SBO-EI algorithm on two of the 24 BBOB functions. These two functions (Rastrigin, Sharp Ridge) were chosen because they nicely represent the main characteristics and problems that can be observed for both infill criteria. These characteristics will be explained in detail in the following paragraphs. A simple random search algorithm was included as a baseline comparison. The convergence plots of all remaining functions are available as supplementary material, also included are the results with the R-package 'mlrmbo'.

As expected, the random search is largely outperformed on three to ten-dimensional function instances. In the two-dimensional instances, the random search approaches the performance of SBO-PV, at least for longer algorithm runtimes. For both functions shown in Figure 1, it can be observed that SBO-EI works as good or even better than SBO-PV on the two-, and three-dimensional function instances. As dimensionality increases, SBO-PV gradually starts to overtake and then outperform SBO-EI.

Initially, SBO-PV seems to have a faster convergence speed on both functions. Yet, this speedup comes at the cost of being more prone to getting stuck in local optima or sub-optimal regions. This problem is most obvious in the two- and three-dimensional instances, especially on the two-dimensional instance of BBOB function 3 (Separable Rastrigin). Here, SBO-EI shows a similar convergence to SBO-PV on roughly the first 40 iterations. However, after just 100 iterations, the algorithm appears to get stuck in local optima, reporting only minimal or no progress at all. As Rastrigin is a highly multi-modal function, the results for SBO-PV are not too surprising. At the same time, SBO-EI yields steady progress over all 300 iterations, exceeding the performance of SBO-PV.

Yet, this promising performance seems to change on higher dimensional functions. On the five- and ten-dimensional function sets, the performance of SBO-PV is close to the one of SBO-EI early on. Later in the run, SBO-PV outperforms SBO-EI noticeably. Neither algorithm shows a similar form of stagnation as was visible for SBO-PV in the lower dimensional test scenarios.

This behavior indicates that with increasing dimensionality, it is less likely for SBO to get stuck in local optima. Therefore, the importance of exploration diminishes with increasing problem dimension. De Ath et al. reach a similar conclusion for the higher dimensional functions they considered [9]. They argue that the comparatively small accuracy that the surrogates can achieve on high dimensional functions results in some internal exploration even for the strictly exploitative SBO-PV. This is due to the fact that the estimated location for the function optimum might be far away from the true optimum. In Section 3.3 this is covered in more detail. There, we investigate exactly how much exploration is done by each infill criterion.

3.2 Statistical Analysis

To provide the reader with as much information as possible in a brief format, the rest of this section will present data that was aggregated via a statistical analysis. We do not expect that our data follows a normal distribution. For example, we know that we have a fixed lower bound on our performance values. Also, our data is likely heteroscedastic (i.e., group variances are not equal). Hence, common parametric test procedures that assume homoscedastic (equal variance), normal distributed data may be unsuited.

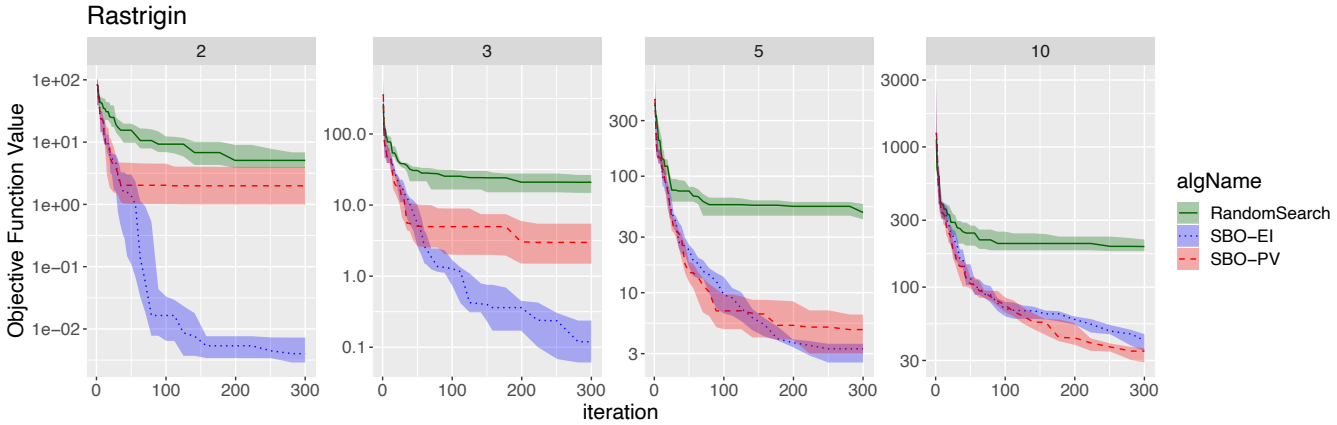
Therefore, we apply non-parametric tests that make less assumptions about the underlying data, as suggested by Derrac et al. [11]. We chose the Wilcoxon test, also known as Mann-Whitney test [19], and use the test implementation from the base-R package 'stats'. Statistical significance is accepted if the corresponding p-values are smaller than $\alpha = 0.05$. The statistical test is applied to the results of each iteration of the given algorithms. As the BBOB suite reports results in exponentially increasing intervals, the plot in Figure 2 follows the same exponential scale on the x-axis. The figure shows the aggregated results of all tests on all functions and iterations. Blue cells indicate that SBO-EI significantly outperformed SBO-PV, while red indicates the opposite result. Uncolored cells indicate that there was no evidence for a statistically significant difference between the two competing algorithms. The figure is further split by the input dimensionality (2,3,5,10) of the respective function, which is indicated on the right-hand side of each subplot.

We start with an overview of the two-dimensional results. Initially, SBO-PV seems to perform slightly better than its competitor. Until roughly iteration 60, it shows statistical dominance on more functions than EI. Yet, given more iterations, EI performs well on more functions than PV. The performance of SBO-PV with less than 60 iterations, together with the convergence plots, indicates that SBO-PV achieves faster convergence rates. Thus, SBO-PV is considered the more greedy algorithm.

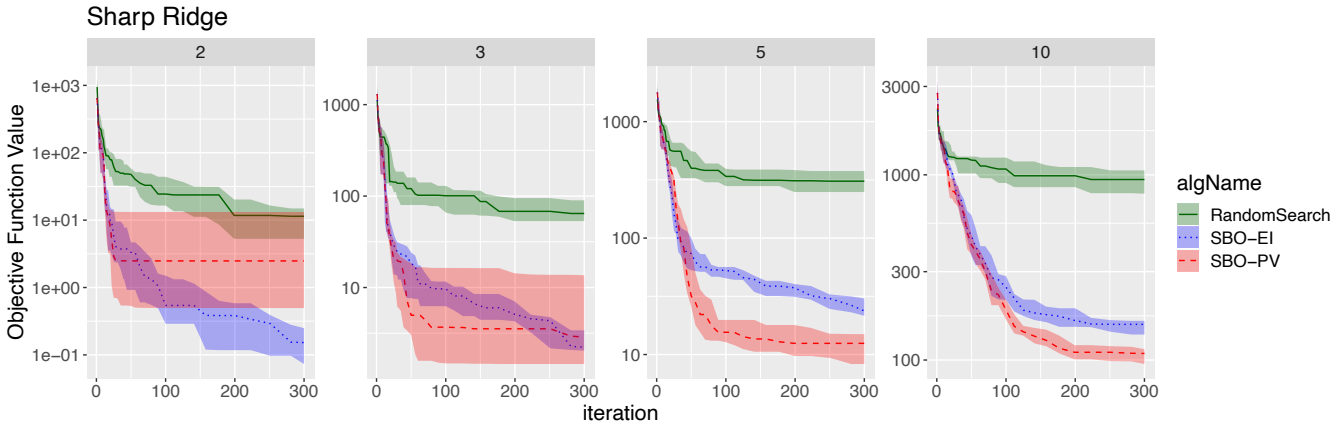
This same greediness of SBO-PV may increasingly lead to stagnation when more than 60 iterations are performed. Here, the SBO-PV algorithm is getting stuck at solutions that are sub-optimal or only locally optimal. SBO-EI starts to outperform SBO-PV, overtaking it at roughly 70 iterations. At the maximum budget of 300 function evaluations, SBO-EI outperforms SBO-PV on 8 of the 24 test functions. SBO-PV only works best on 3 out of the 24 functions for the two-dimensional instances (Sphere, Linear Slope, Rosenbrock). Notably, those three functions are unimodal (at least for the two-dimensional case discussed so far). It is not surprising to see the potentially more greedy SBO-PV perform well on unimodal functions.

A similar behavior is observed on the three-dimensional function set. Initially, SBO-PV performs well on up to five functions, while SBO-EI only performs better on up to two functions. At around 85 iterations, SBO-EI again overtakes SBO-PV and then continues to perform well on more than eight functions up to the maximum budget of 300 iterations.

Based on the observations for the two- and three-dimensional scenarios, one could assume that a similar pattern would be observed for higher-dimensional functions. However, as previously discussed, it seems that SBO-PV's convergence rate is less likely



(a) BBOB Function 3 - Separable Rastrigin



(b) BBOB Function 13 - Sharp Ridge

Figure 1: Convergence plots of SBO-EI and SBO-PV on two of the 24 BBOB functions. Random search added as baseline. The experiments are run (from left to right) on 2,3,5, and 10 input-dimensions. The y-axis shows the best so far achieved objective function value on a logarithmic scale. The center line indicates the median of the algorithm repeats. The surrounding ribbon marks the lower and upper quartiles. Y-values measure differences from the global function optimum.

to stagnate with increasing problem dimensionality. On the five-dimensional functions, only three functions remain on which SBO-EI outperforms at the maximum given budget. Namely: Rastrigin, Büche-Rastrigin, and Gallagher's Gaussian 101-me peaks function. All of which have a large number of local optima. Furthermore, Gallagher's function has little to no global structure, thus requiring a more explorative search. On the other hand, SBO-PV performs better on up to 7 functions. This now also includes multimodal functions, hence functions that are usually not considered promising candidates for SBO-PV.

On the ten-dimensional function set, SBO-EI is outperformed on nearly all functions, with only two temporal exceptions. Only on the Sphere function and the Katsuura function, a statistically significant difference can be measured for a few iterations in favour of SBO-EI. SBO-PV performs significantly better on 9 out of the 24 functions. Only on the function group with weak global structure, SBO-PV fails to produce significantly better performance.

Summarizing these results for separate function groups, it is noticeable that SBO-EI tends to work better on the multimodal functions. Here, SBO-EI clearly excels on the two-, and three-dimensional instances. On the functions with weak global structure, it continues to excel on the five-dimensional functions and is at least not outperformed on the ten-dimensional ones.

SBO-EI performs especially poor on 'functions with low or moderate conditioning'. Here, SBO-PV outperforms SBO-EI on at least as many functions as vice versa, independent of the budget and the problem dimensionality. Generally, multimodality does not seem to require an explorative search methodology as long as the input dimensionality is high.

3.3 Case Study: Measuring Exploration

The performance discussion in Section 3.1 largely relies on the idea that SBO-PV generally does less exploration than SBO-EI.

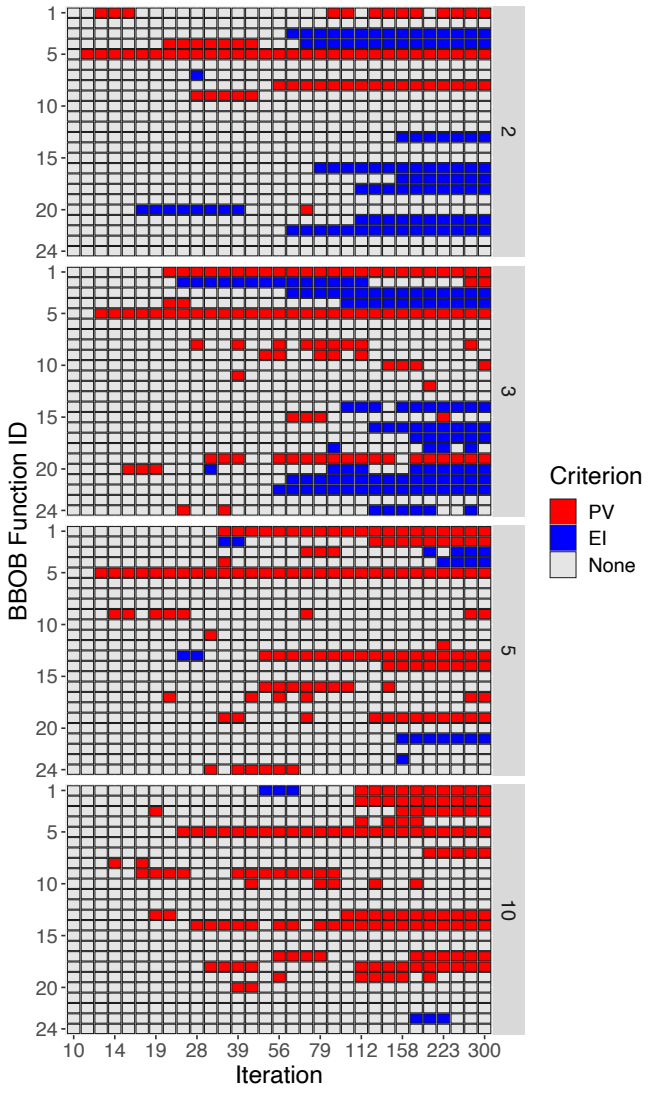


Figure 2: Statistical analysis of difference in performance between SBO-EI vs. SBO-PV. y-axis shows the BBOB-function index. Colors indicate which infill criterion performed better in each case, based on the statistical test procedure. Blank boxes indicates that there was no significant difference found. The results are presented for the 2,3,5 and 10 dimensional BBOB-function set as indicated in the gray bar to the right of each plot segment. The BBOB suite reports results in exponentially increasing intervals, thus the figure follows the same exponential scale on the x-axis.

Multiple functions were observed on which the search with SBO-PV stagnates too early (cf. Section 3.1). On the other hand, we argue that as long as SBO-PV does not get stuck in sub-optimal regions, its greedy behavior leads to faster convergence and, thus, a more efficient search process. This may especially be true if the budget is small in relation to the problem dimension. In other words, if time is short, greedy behavior may be preferable.

To support this hypothesis, additional experiments were carried out to determine the amount of "exploration" each criterion does in different stages of the optimization runs. For this purpose, we assume that exploration can be estimated as the Euclidean distance of a proposed candidate to its closest neighbor in the already evaluated set of candidate solutions. Therefore, placing a new point close to a known point is regarded as an exploitative step. Else, placing a new point far away from known points is considered as an explorative step.

This measure was applied to the previously discussed optimization runs on the BBOB function set. The results for the two functions are shown in Figure 3. Represented are the same functions as in the previous section for comparability. Results of the case study on all 24 BBOB functions are included in the supplementary material.

As discussed earlier, the performance of SBO-PV stagnated on the two- and three-dimensional instances of this function. Hence, it is interesting to see that the measured distance becomes fairly small in those same cases. While SBO-EI seems to make comparatively large search steps, SBO-PV seems to propose candidate solutions in the close vicinity of already known solutions.

On the higher-dimensional functions, the difference between SBO-EI and SBO-PV decreases. On the Rastrigin function, both infill criteria reach similar levels on the ten-dimensional function. On the Sharp Ridge problem, a significant difference between the criteria remains. But the difference decreases with increasing problem dimension. This supports the idea that the convergence speed of SBO-PV is generally higher due to less time being spent on exploring the search space.

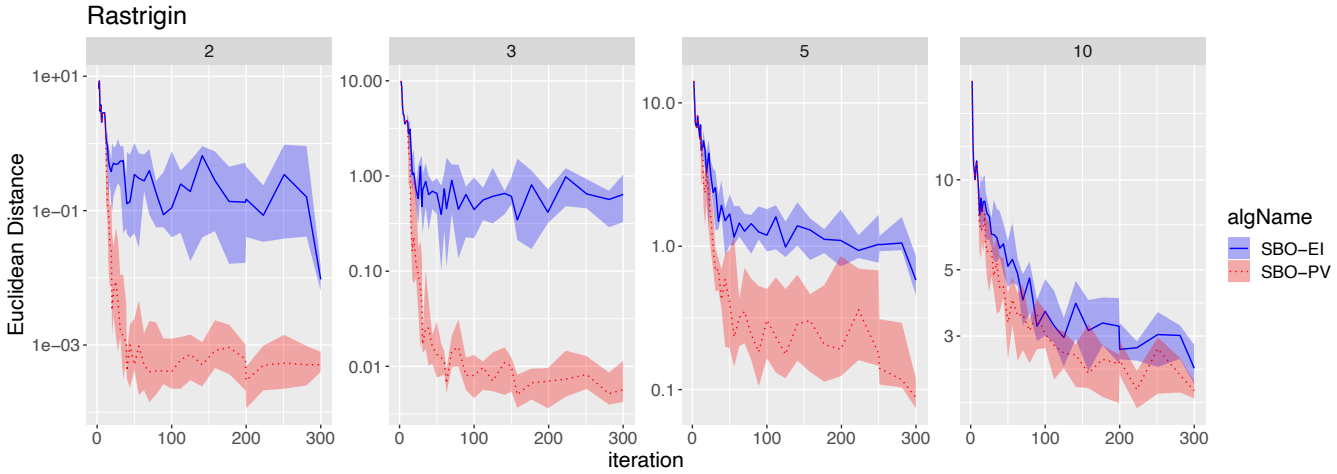
4 DISCUSSION

Before giving a final conclusion and an outlook on possible future work, we will reconsider the underlying research questions of this investigation.

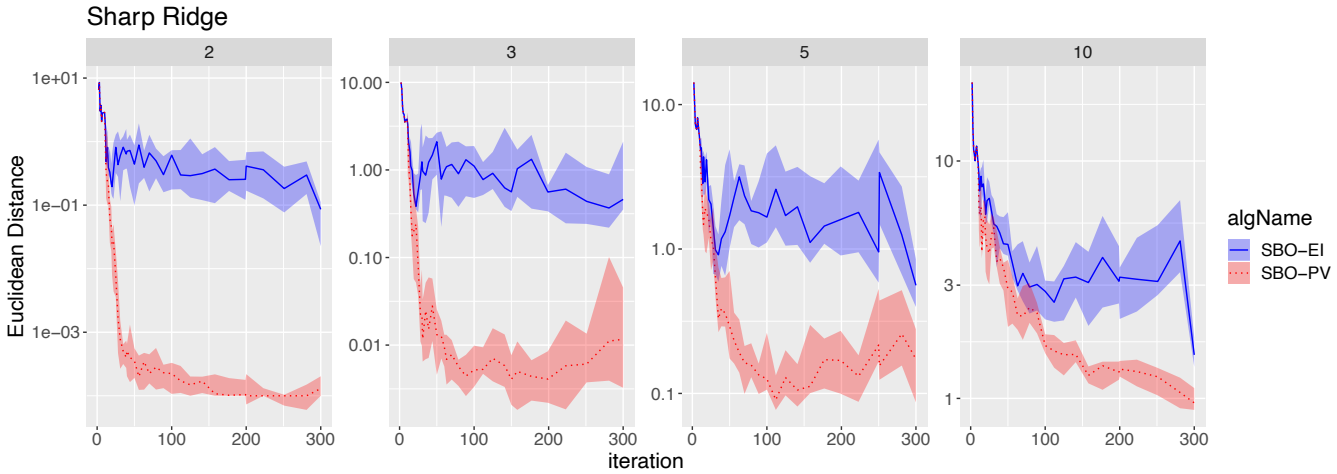
RQ-1 Can distinct scenarios be identified where PV outperforms EI or vice versa?

The large quantity of results measured on the BBOB and the smoof function sets (cf. Section 2.3) allows identifying scenarios in which either PV or EI perform best.

- **Problem dimension:** Considering the previously presented results, SBO-EI performs better on lower-dimensional functions, whereas SBO-PV excels on higher dimensional functions. If no further knowledge about a given optimization problem is available, then the benchmark results indicate that it is best to apply SBO-EI to functions with up to three dimensions. Problems with five or more input dimensions are likely best solved by SBO-PV.
- **Budget:** The main drawback of an exploitative search (SBO-PV) is the likelihood of prematurely converging into a local optimum. The larger the available budget of function evaluations, the more likely it is that SBO-PV will converge to a local optimum. Yet, as long as SBO-PV is not stuck, it is more efficient in finding the overall best objective function value. Therefore, for many functions, there should be a certain *critical budget* until which SBO-PV outperforms SBO-EI.



(a) BBOB Function 3 - Separable Rastrigin



(b) BBOB Function 13 - Sharp Ridge

Figure 3: The proposed candidate solutions' distance to its nearest neighbor in the set of known solutions. The y-axis shows the measured Euclidean distance. The experiments are run (from left to right) on 2,3,5, and 10 input-dimensions. The colored lines indicates the median of the distance, computed over the repeated runs of each algorithm. The surrounding ribbon marks the lower and upper quartiles.

If the budget is larger, SBO-EI performs better. The results indicate that this *critical budget* is increasing with input dimensionality. In the discussed benchmarks, SBO-EI started to excel after roughly 70 iterations on the two- and three-dimensional functions. On the five-dimensional set, SBO-EI started excelling on some functions at around 200 iterations. On the ten-dimensional, no signs of approaching such a *critical budget* could be observed, indicating that it might lie far beyond (i.e. $\gg$ 300 evaluations) reasonable budgets for SBO.

- **Modality:** Lastly, any a priori knowledge regarding the landscape of the given function can be used to make a more informed decision. For this, the function classes and landscape features of the BBOB function should be considered. The greatest weakness of SBO-PV is to get stuck in local optima (or flat, sub-optimal regions). Hence, if it is known that a given function is fairly multimodal, then EI may be a good choice. For simpler, potentially unimodal functions, we recommend using PV.

RQ-2 Is EI a reasonable choice as a default infill criterion?

Since the best infill criterion changes depending on the optimization problem, a simple answer to this question can not be given. We suggest that the default choice should be determined depending on what is known about the use case.

- **Problem dimension:** As discussed above, problem dimension is a crucial factor. Even when optimizing a black-box function, the input dimensionality of the function should be known a priori. Therefore, being able to select a proper infill criterion based on input dimensionality should be applicable

to most optimization tasks. The problem dimension is usually provided to the algorithm when the user specifies the search bounds.

- **Budget:** The available budget of function evaluations can be considered for the selection of a good infill criterion. As discussed above, the number of evaluations has a strong impact on whether the EI or the PV criterion performs better. However, the budget can not be determined automatically and will depend on fairly problem-specific knowledge. In practice, the importance of this has to be communicated to practitioners.
- **Modality:** Another impact factor is the modality of the search landscape. For black-box problems, this is usually not known. To avoid premature convergence, a conservative choice would be to select EI (as long as the problems are low-dimensional and the budget is relatively high).
- **Criterion Complexity:** Regardless of performance, there may be an additional argument for selecting the PV criterion: its simplicity. This has two consequences. Firstly, it is easier to implement and compute. Secondly, it is easier to explain, which may help to bolster the acceptance of SBO algorithms in practice.

5 CONCLUSION

In conclusion, we observe that SBO-EI performs differently than often assumed. Especially on higher-dimensional functions, SBO-PV seems to be the better choice, despite (or because) of its fairly greedy search strategy. Only on lower-dimensional, multimodal functions, SBO-PV is more likely to get stuck in local optima, at least given the limited budgets under which our benchmark study was performed. In these cases, the largely explorative approach of SBO-EI is required to escape local optima. The proposed case study confirmed that the estimated exploration between SBO-PV and SBO-EI, as measured in distance to the nearest neighbor, decreases with increasing dimensionality.

Finally, we would like to end this paper with suggestions for future research. First and foremost, the experiments clearly show that the PV is a surprisingly competitive infill criterion. Future work on new infill criteria should include PV in their benchmarks (e.g., as a baseline). Portfolio methods that cover multiple infill criteria would likely profit from considering PV in their framework.

Future work should reassess the importance of explorative SBO in practical applications. In this context, the sub-optimal performance of SBO-EI for dimensionalities of five or more is troubling and needs to be further investigated.

A consideration that was not covered in this work, is the global model quality. A globally accurate model is not required for an optimization task that only searches for a single optimum. However, there may be additional requirements in practice. For instance, the learned model may have to be used after the optimization run to provide additional understanding of the real-world process to practitioners or operators. A model that is trained with data generated by a purely exploitative search might fall short of this requirement, despite its ability to find a good solution.

REFERENCES

[1] Peter Auer. 2002. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* 3, Nov (2002), 397–422.

[2] T. Bartz-Beielstein, C.W.G. Lasarczyk, and M. Preuss. 2005. Sequential parameter optimization. In *2005 IEEE Congress on Evolutionary Computation (CEC 2005)*, Vol. 1. IEEE, Edinburgh, Scotland, UK, 773–780.

[3] Thomas Bartz-Beielstein, Joerg Stork, Martin Zaefferer, Margarita Rebolledo, Christian Lasarczyk, Joerg Ziegenhirt, Wolfgang Konen, Oliver Flasch, Patrick Koch, Martina Friese, Lorenzo Gentile, and Frederik Rehbach. 2019. SPOT: Sequential Parameter Optimization Toolbox – Version 2.0.4. Online. (2019). Available: https://cran.r-project.org/package=SPOT, accessed: 2019-11-15.

[4] Bernd Bischl, Jakob Richter, Jakob Bossek, Daniel Horn, Michel Lang, and Janek Thomas. 2019. mlrMBO: Bayesian Optimization and Model-Based Optimization of Expensive Black-Box Functions – Version 1.1.2. Online. (2019). Available: https://cran.r-project.org/package=mlrMBO, accessed: 2019-11-15.

[5] Bernd Bischl, Jakob Richter, Jakob Bossek, Daniel Horn, Janek Thomas, and Michel Lang. 2017. *mlrMBO: A Modular Framework for Model-Based Optimization of Expensive Black-Box Functions.* http://arxiv.org/abs/1703.03373

[6] Jakob Bossek. 2017. smoof: Single- and Multi-Objective Optimization Test Functions. *The R Journal* (2017). https://journal.r-project.org/archive/2017/RJ-2017-004/index.html

[7] Jakob Bossek and Pascal Kerschke. 2017. smoof: Single and Multi-Objective Optimization Test Functions – Version 1.5.1. Online. (2017). Available: https://cran.r-project.org/package=smoof, accessed: 2019-11-15.

[8] Adam D. Bull. 2011. Convergence Rates of Efficient Global Optimization Algorithms. *Journal of Machine Learning Research* 12 (Oct. 2011), 2879–2904.

[9] George De Ath, Richard M. Everson, Alma A. M. Rahat, and Jonathan E. Fieldsend. 2019. Greed is Good: Exploration and Exploitation Trade-offs in Bayesian Optimisation. *ArXiv e-prints* (2019). arXiv:http://arxiv.org/abs/1911.12809v1

[10] Thiago de P. Vasconcelos, Daniel A. R. M. A. de Souza, César L. C. Mattos, and João P. P. Gomes. [n. d.]. No-PASt-BO: Normalized Portfolio Allocation Strategy for Bayesian Optimization. ([n. d.]). arXiv:http://arxiv.org/abs/1908.00361v1

[11] Joaquín Derrac, Salvador García, Daniel Molina, and Francisco Herrera. 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation* 1, 1 (March 2011), 3–18. https://doi.org/10.1016/j.swevo.2011.02.002

[12] Alexander Forrester, Andras Sobester, and Andy Keane. 2008. *Engineering Design via Surrogate Modelling.* Wiley, New York, NY.

[13] David Ginsbourger and Rodolphe Le Riche. 2010. Towards Gaussian Process-based Optimization with Finite Time Horizon. In *mODa 9 – Advances in Model-Oriented Design and Analysis*, Alessandra Giovagnoli, Anthony C. Atkinson, Bernard Torsney, and Caterin May (Eds.). Physica-Verlag HD, Bertinoro, Italy, 89–96. https://doi.org/10.1007/978-3-7908-2410-0_12

[14] Nikolaus Hansen, Anne Auger, Olaf Mersmann, Tea Tusar, and Dimo Brockhoff. 2016. COCO: A Platform for Comparing Continuous Optimizers in a Black-Box Setting. *ArXiv e-prints* (Aug. 2016). ArXiv ID: 1603.08785v3.

[15] Nikolaus Hansen, Dimo Brockhoff, Olaf Mersmann, Tea Tusar, Dejan Tusar, Ouassim Ait ElHara, Phillipe R. Sampaio, Asma Atamna, Konstantinos Varelas, Umut Batu, Duc Manh Nguyen, Filip Matzner, and Anne Auger. 2019. COmparing Continuous Optimizers: numbbo/COCO on Github. (March 2019). https://doi.org/10.5281/zenodo.2594848

[16] Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. 2009. *Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions.* Technical Report RR-6829. INRIA.

[17] José Miguel Hernández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani. 2014. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*, Vol. 1. MIT Press, Cambridge, MA, USA, 918–926. http://dl.acm.org/citation.cfm?id=2968826.2968929

[18] Matthew Hoffman, Eric Brochu, and Nando de Freitas. 2011. Portfolio Allocation for Bayesian Optimization. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI'11).* AUAI Press, Arlington, Virginia, United States, 327–336. http://dl.acm.org/citation.cfm?id=3020548.3020587

[19] Myles Hollander, Douglas A. Wolfe, and Eric Chicken. 2014. *Nonparametric Statistical Methods* (3rd ed.). Wiley, New York, NY.

[20] Donald R. Jones, Matthias Schonlau, and William J. Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization* 13, 4 (Dec. 1998), 455–492. https://doi.org/10.1023/A:1008306431147

[21] Remi Lam, Karen Willcox, and David H. Wolpert. 2016. Bayesian Optimization with a Finite Budget: An Approximate Dynamic Programming Approach. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 883–891.

[22] Michael D McKay, Richard J Beckman, and William J Conover. 1979. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* 21, 2 (1979), 239–245.

[23] Vu Nguyen, Santu Rana, Sunil K Gupta, Cheng Li, and Svetha Venkatesh. 2016. Budgeted Batch Bayesian Optimization. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE. https://doi.org/10.1109/icdm.2016.0144

[24] Umberto Noé and Dirk Husmeier. 2018. On a New Improvement-Based Acquisition Function for Bayesian Optimization. (2018). arXiv:http://arxiv.org/abs/1808.06918v1

[25] Michael Osborne. 2010. *Bayesian Gaussian processes for sequential prediction, optimisation and quadrature*. Ph.D. Dissertation. University of Oxford.

[26] Olivier Roustant, David Ginsbourger, and Yves Deville. 2012. DiceKriging, DiceOptim: Two R Packages for the Analysis of Computer Experiments by Kriging-Based Metamodeling and Optimization. *Journal of Statistical Software* 51, 1 (2012), 1–55. http://www.jstatsoft.org/v51/i01/

[27] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.). Curran Associates, Inc., Lake Tahoe, NV, USA, 2951–2959.

[28] Rainer Storn and Kenneth Price. 1997. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization* 11, 4 (1997), 341–359.

[29] Bas van Stein, Hao Wang, Wojtek Kowalczyk, Thomas Bäck, and Michael Emmerich. 2015. Optimally Weighted Cluster Kriging for Big Data Regression. In *Advances in Intelligent Data Analysis XIV, 14th International Symposium, IDA 2015 (Lecture Notes in Computer Science)*, Elisa Fromont, Tijl De Bie, and Matthijs van Leeuwen (Eds.), Vol. 9385. Springer, Saint Etienne, France, 310–321. https://doi.org/10.1007/978-3-319-24465-5_27

[30] Hao Wang, Michael Emmerich, and Thomas Back. 2018. Cooling Strategies for the Moment-Generating Function in Bayesian Global Optimization. In *2018 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, Rio de Janeiro, Brazil, 1–8. https://doi.org/10.1109/cec.2018.8477956

[31] Hao Wang, Michael Emmerich, and Thomas Bäck. 2019. Towards self-adaptive efficient global optimization. In *Proceedings LeGO – 14th International Global Optimization Workshop (AIP Conference Proceedings)*, Vol. 2070. Leiden, NL. https://doi.org/10.1063/1.5090023

[32] Hao Wang, Bas van Stein, Michael Emmerich, and Thomas Bäck. 2017. Time complexity reduction in efficient global optimization using cluster Kriging. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'17)*. ACM, Berlin, Germany, 889–896. https://doi.org/10.1145/3071178.3071321

[33] Simon Wessing and Mike Preuss. 2017. The true destination of EGO is multi-local optimization. In *2017 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. IEEE. https://doi.org/10.1109/la-cci.2017.8285677