Boosting Parameter-Tuning Efficiency with Adaptive Experimental Designs

Jörg Stork*, Andreas Fischbach*, Thomas Bartz-Beielstein*, Martin Zaefferer*, A.E. Eiben †

1 Introduction

Model-based tuning has proven to be a successful method for improving the performance of computational intelligence methods such as evolutionary algorithms or neural networks [1, 2, 3]. However, model-based tuning itself can be a demanding and time consuming task. One crucial step during the tuning process is the selection of an adequate experimental design as well as the limits of the algorithm parameter space to be explored, the so-called *region of interest* (ROI). In the current practice, the ROI is static, that is, chosen *a priori* and not changed during the tuning process.

In this paper we will investigate adaptive ROIs. In particular, we introduce mechanisms for appropriately locating and sizing the ROI on-the-fly. We will focus on the sizing aspects, because too large ROIs may slow down the tuning process resulting in worse results. This is due to a large search space leading to a lack of detail in the most critical regions. The meta model would not be able to represent these regions adequately. By adapting the size of the ROI during the tuning process (online) this issue can be dealt with.

To understand the working principles of adaptive ROIs, we implement special *moving* and *zooming* operations, where *zooming* can make the ROI smaller (*zooming in*) or larger (*zooming out*). Furthermore, we distinguish four algorithm variants, depending on whether the model building and the sampling steps utilize (or not) the adapted ROI settings, i.e., the zoomed, local information. Here by we obtain four main variants and our primary research question is: What is the effect of these policies on the tuning process?

Our approach to answer this question is experimental. Our adaptive ROIs are tested with simple benchmark functions and we analyze the effect of

*Dept. of Comp. Sci. and Eng. Sci., Cologne University of Applied Sciences, `http://www.spotseven.de`, E-Mail: firstname.lastname@fh-koeln.de

†Computational Intelligence Group, Dept. of Comp. Sci., Vrije Universiteit Amsterdam, E-Mail: a.e.eiben@vu.nl

the moving and zooming operations on tuner performance. The rest of this paper is organized as follows. Section 2 describes previous research in this field. The methods employed in this study are introduced in Sec. 3. This includes the employed tuning software as well as the ROI adaptation methodology. A detailed description of the experimental setup is given in Sec. 4. Afterwards, results are presented in Sec. 5. Finally, Sec. 6 concludes this work and gives a short outlook on promising future directions of research.

2 Previous Research

Adaptation of optimization bounds is not a totally new idea. It is closely related to concepts used in adaptive *evolution strategies* (ES). The most simple ES, the so called (1+1)-ES, has a step-size parameter which is multiplied with a preset factor in case of optimization failure, and divided by that factor in case of success [4]. Similar and more complex adaptations of step-sizes or mutation-rates can found in most optimization algorithms.

While model-based tuning algorithms such as *sequential parameter optimization* (SPO), see 3.1) do also employ such optimization algorithms when optimizing the model, they do not necessarily do so on the meta-level. Here, the step size may be best translated to the region in which the search is performed.

One similar approach is often employed with linear models, i.e., in the *response surface methodology* (RSM) [5]. Here, it is obvious that restricting a model to a certain region rather than exploiting knowledge of the whole search space makes sense. A linear model may not fit a complex function on its global scale. Also, depending on the specific model, a global optimization of it would often lead to placing new design points on outmost border of the search space. Hence, RSM adapts the search space sequentially. The question whether this does also make sense with other model-types is one motivation for this work.

A related approach has been introduced and further studied in [6, 7, 8]. The REVAC method is implicitly changing the ROI by continually adapting a distribution used to sample the parameter space. It has been shown to be effective in optimizing already highly developed evolutionary algorithms [9].

3 Methods

3.1 Sequential Parameter Optimization

The performance of most optimization algorithms is influenced by several parameters. Choosing those parameters in a meaningful and beneficial way may be difficult, especially considering that an ideal choice depends on the specific problem to be solved by the optimizer. The meta optimization of an algorithms parameters is often referred to as tuning. Tuning yields not only improved algorithms, but also allows for a fair comparison between competing algorithms.

One framework developed for parameter tuning is SPO [10] . It has been applied to numerous applications [11]. In its core, it is uses methods such as *design of experiments* (DoE), statistics, and machine learning.

A typical SPO run begins with the generation of an initial design. That is, several configurations of different parameter values are generated, e.g., with methods like Latin Hypercube Sampling. The design is generated in an predefined area of the search space, as defined by the ROI. The initial design is then evaluated with the tuned optimization algorithm to determine the quality of the chosen parameter values. With the gained knowledge about each configurations quality, a surrogate model is learned, which is supposed to represent how the parameters influence the quality.

Next, this surrogate model itself is subject to an optimization process. The best parameter configuration (according to the model) is determined. This again takes place in the ROI, and may be based on any algorithm including classical optimization algorithms, evolutionary algorithms or sampling methods (DoE). The optimal solution found may then be evaluated with the tuned algorithm. These steps of model building, model optimization, and algorithm evaluation are repeated in a sequential manner until some specified stopping criterion (e.g., number of steps, number of evaluations) is reached.

3.2 Adaptive ROI

The ROI defines lower and upper limits in each search space dimension and is usually set by an experienced user with more or less good knowledge or at least ideas of the interesting regions of the fitness landscape. These ROI limits are hard limits. The adaptation is performed in each sequential step. In case of the first iteration, it takes place directly after the evaluation of the initial points, which are given by either a Latin hypercube sampling or a random uniform distributed sample. In the sequential iterations, adaptation takes place directly after evaluating the most recent candidate solution on the target function.

The adaptation can be controlled by two parameters. The first parameter (**MOVE**) controls whether the ROI is moved towards good candidate points or not. The second parameter (**z**) controls the zooming factor of the ROI. The ROI expansion will not go beyond the user defined ROI limits. Both operations, move (if enabled) and zooming are performed in each step. That means, the ROI will be adapted in each iteration by moving, extending, or shrinking the former ROI range. The range is hereby calculated as the distance between the upper an lower limit or the ROI. There are two cases in which different actions are performed by the adaptation (algorithm 1):

1. **Improvement.** If the new best point leads to an improvement, which means it is not equal to the last found point, and the move operation is enabled, than the center of the ROI is moved to this new point in the next iteration. Also, depending on factor **z**, the ROI is either shrinked ($\mathbf{z} > 1$) or extended ($\mathbf{z} < 1$). The first adaptation will be referred to as *zooming in*, whereas the latter is referred to as *zooming out*.

2. **Stagnation.** The center is not moved and the ROI range is either shrunk ($\mathbf{z} < 1$) or extended ($\mathbf{z} > 1$)

This procedure relies on the assumption that optima are situated in the neighborhood of good solutions. For **z** values larger than 1.0, the search is localized by shrinking the ROI if an improvement is achieved. And, if no improvement is made, the ROI is extended to be able to escape local optima.

The case of **z** smaller than 1.0 may yield a more localized, exploitative search in case of stagnation. This could help to focus more on the immediate neighborhood of the best solution. In contrast to the above case, one would not try to escape the local optimum, but rather to improve the best solution found so far in a more local sense. At the same time, improvement may yield a larger ROI, hence progress may be sped up by allowing an even larger step in the next SPO iteration.

Which strategy is better is supposed to be determined in the experiments.

Algorithm 1 Adaptation by Moving and Zooming

```
if $newCandidatePoint < bestFitness$ then
    $w \leftarrow 1/z$                              ▷ Zooming(-in) ROI
else
    $w \leftarrow z$                                ▷ Zooming(-out) ROI
end if
$l \leftarrow b - a$                                ▷ Compute length of ROI
if MOVE then
    $p \leftarrow newCandidatePoint$                ▷ Move to new best x position
else
    $p \leftarrow a + l/2$
end if
$a \leftarrow p - w \times l/2$
$b \leftarrow p + w \times l/2$
if then$((b - a) < 1e - 20)$                        ▷ If necessary reset ROI
    $a \leftarrow a0$
    $b \leftarrow b0$
end if
```

The above notions describe in detail when and how the adaptation is performed. One additional open question is how an adapted ROI should be used. That is, it is unclear whether the ROI should affect the boundaries of the search on the surrogate model, the selection of points for training the surrogate model, or even both.

4 Experimental Setup

4.1 Designed Experiments

Adaptation and moving is closely related to locality. A series of experiments was performed to answer the question how local or global model building affect the performance. Prediction of new points based on meta models is done in two steps: first, a (sub)set of points is chosen for the model building. Then, a second set of points has to be selected for the predictions. The correct method is of crucial importance for the search process. Therefore, four possible combinations of building and prediction are included in the design. The corresponding design parameters are explained in Sect. 4.3.

4.2 Objective Functions

The function f_1 is used to demonstrate positive aroi effects.

$$f_1(x) = \begin{cases} 1 & x < -1 \\ x^2 & \text{otherwise.} \end{cases} \tag{1}$$

The function f_2, also known as the *wild* function, is defined as follows.

$$f_2(x) = 10\sin(0.3x)\sin(1.3x^2) + 10^{-5}x^4 + 0.2x + 80. \tag{2}$$

The minimum is located at $x_{opt} = -15.81515$ with $f_2(x_{opt}) = 67.46773$.

Rastrigin's function, which is defined as

$$f_3(x) = 10 + x^2 - 10\cos(2\pi x) \tag{3}$$

was chosen as the third test function. The minimum is located at $x_{opt} = 0$ with $f_3(x_{opt}) = 0$.

All three functions are visualized in Fig. 1.

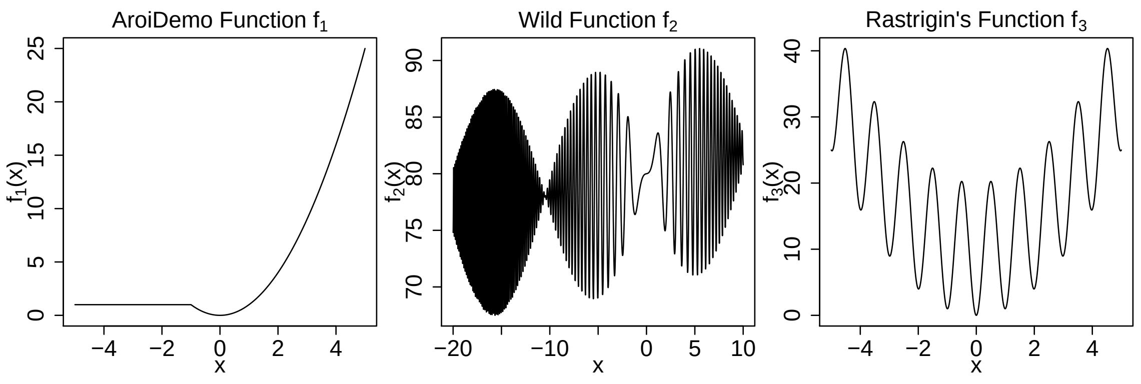

Figure 1: Visualization of the three objective functions.

4.3 Design

The initial design size, n, describes the number of initial sample points, which are evaluated to build the first meta model. Initial design sizes of five and ten were chosen for our experiments.

Regarding the meta modeling, there are four case to be considered.

Mdl-1 (global,global): meta model built on global data, predictions sampled in the region of the global data, i.e., $a(t) = a0$ and $b(t) = b0$ in every time step. Both zooming, z, and moving, $MOVE$, have no effect in this case.

Mdl-2 (global, local): meta model built on global data, predictions sampled in the region of the local data. Here, the samples are taken from the adapted ROI settings $a(t)$ and $b(t)$. The model is build with all evaluated design points at time step t.

Mdl-3 (local, global): meta model built on local data, predictions sampled in the region of the global data. Here, the model building is based on the x design points, which belong to the current ROI, i.e., $x(t) \in [a(t), b(t)]$.

Mdl-4 (local, local): meta model built on local data, predictions sampled in the region of the local data. Here, the model building is based on the x design points, which belong to the current ROI, i.e., $x(t) \in [a(t), b(t)]$. The samples are taken from the adapted ROI settings $a(t)$ and $b(t)$.

The number of repeats, i.e., the number of algorithm runs with unmodified parameter settings but different random seeds, is denoted as nRepeats.

4.4 Algorithm Parameter

Parameters of Algorithm 1 were chosen as follows: Number of points evaluated on the meta model $m = 1000$. To generate points on the meta model, the function `maximinLHS()` was used.

Experimental setups used in this study are shown in Tab. 1.

Table 1: Experimental Setup

	exp1	exprg05	rast01	exprg07	exprg08	exprg09	exprg10
nRepeats	5	50	10	100	100	100	100
zSeq	zSeq1	zSeq1	zSeq1	zSeq2	zSeq3	zSeq2	zSeq2
n	(5,10)	(5,10)	(5,10)	5	5	5	5
MOVE	(T,F)	(T,F)	(T,F)	(T,F)	(T,F)	(T,F)	(T,F)
iter	10	10	10	10	10	10	10
roiMDL		Mdl1-4		Mdl1-4	Mdl1-4	Mdl1-4	Mdl1-4
a0	-10	aSeq	-10	-1	-10	-20	-5.12
b0	10	10	10	10	1	10	15.12
fNum	01	03	01	01	02	03	

Settings for the zoom parameter z are chosen from the set $zSeq1$ = { 0.1, 0.9, 1,1.1, 2.0}, $zSeq2$ = { 0.1, 0.2, ...,1.9, 2.0}, $zSeq3$ = c(0.1, 0.2, ..., 4.9, 5.0). $aSeq$ = { -10, -9, ..., 0, 1}. The sample size is set to m =1000 and a LHD was chosen for each setting. Moving was controlled by the parameter M with values from $\{true, false\}$. The parameter $iter$ denotes the number of function evaluations after the initial design was evaluated. The total number of function evaluations per run can be determined as $n + iter$.

5 Results

5.1 Effect of Moving and Shrinking

The first experiments are labeled *exp1*, *rast01*, and *wild01*. The initial ROI $a = -10, b = 10$ leads to the result shown in Fig 2. At the first sight, shrinking in case of stagnation leads to better results: the smallest function values were obtained with $z = 0.1$. Furthermore, moving of the ROI center point worsens the performance.

Note, this effect can be explained as follows. This set of experiments uses ROIs, which are symmetric to the center, the location of the optimum. Simply shrinking the interval, regardless of further considerations, obviously improves the function value. Since the ROI initially set by the user can not be exceeded, extending the ROI can never get past this bound. On the other hand, any shrinking of the ROI will naturally lead to improved optimization performance. In the most extreme case, the ROI would simply collapse to the location of the optimum itself, with $a0 = b0 = x_{opt}$.

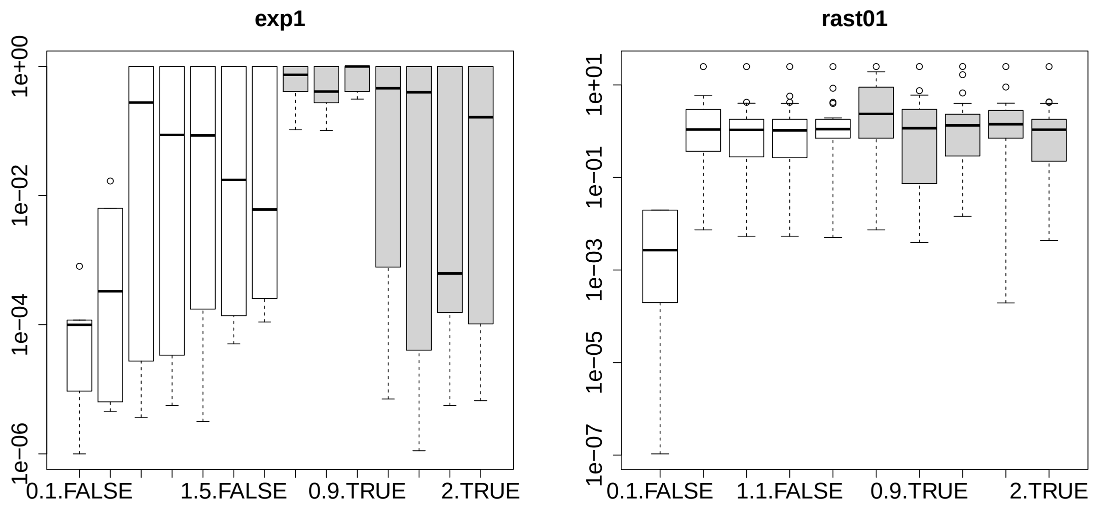

Figure 2: Function values (log.) plotted against $z * M$. Smaller values are better. *Left:* Results from exp1. Leftmost configuration uses M=FALSE, z=0.1. White = no move, gray = move. *Right:* Results from rast01.

To avoid this unwanted side-effect, i.e., optimization is driven by an adequate shrinking procedure and not by the optimization algorithms, a series of experiments were designed, which prevent this behavior. These experiments use asymmetric ROIs, so by simply shrinking the search interval, optimal values cannot be detected.

5.2 Asymmetric Search Interval and Locality in the Meta Model

Interesting effects occur if ROI symmetry is disturbed. In addition to the variations of the ROI locations, experiments which analyze the impact of the adaptation on the meta modeling (as described in Sect. 4.3) are performed. Since four meta modeling variants ($roiGlobal$) were implemented, their comparison might be of interest. The obtained results of the $\log(y)$ value are plotted as a function of the shrinking parameter z and the $roiGlobal$ factor. As can be seen in Fig. 3, local model result in performance improvements when moving is enabled.

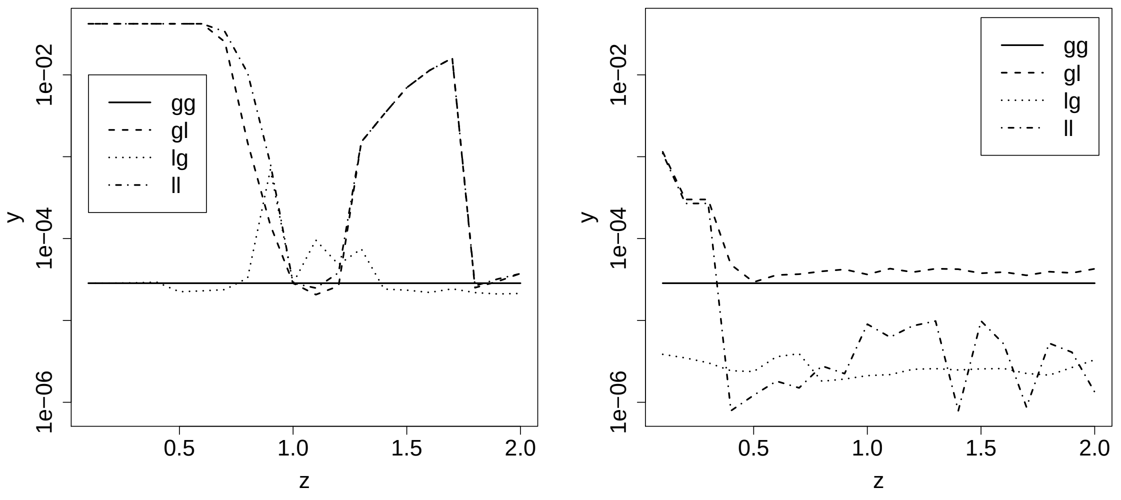

Figure 3: Function values plotted against z for the different modeling approaches; Results from exprg07. *Left:* without moving, a narrow valley can be seen around $z = 1$. *Right:* with moving, the ll and gl settings perform best.

Without Moving (left), the gg setting displays for most of the chosen settings for z an dominating behavior, but in the center a slightly better performance is achieved by the ll and gl model. Furthermore, as the best performance is obtained for values close to $z = 1$, zooming seems to have a negative effect. With moving (right) the local meta modeling performs best, while zooming seems to have no significant effect. This behavior can be explained by a closer look at the function and the algorithm. Due to the asymmetric ROI (-1,10), the optimum is situated at the very left. If the center point cannot be moved, shrinking leads to an exclusion of the optimum. With moving, the center point of the ROI is moved to the left and the range becomes smaller because it is calculated as the difference between the upper and the here lower hard- limit of the ROI. Thus, even without zooming, the adapted ROI is shrinked and a localized search is performed in the region of the optimum.

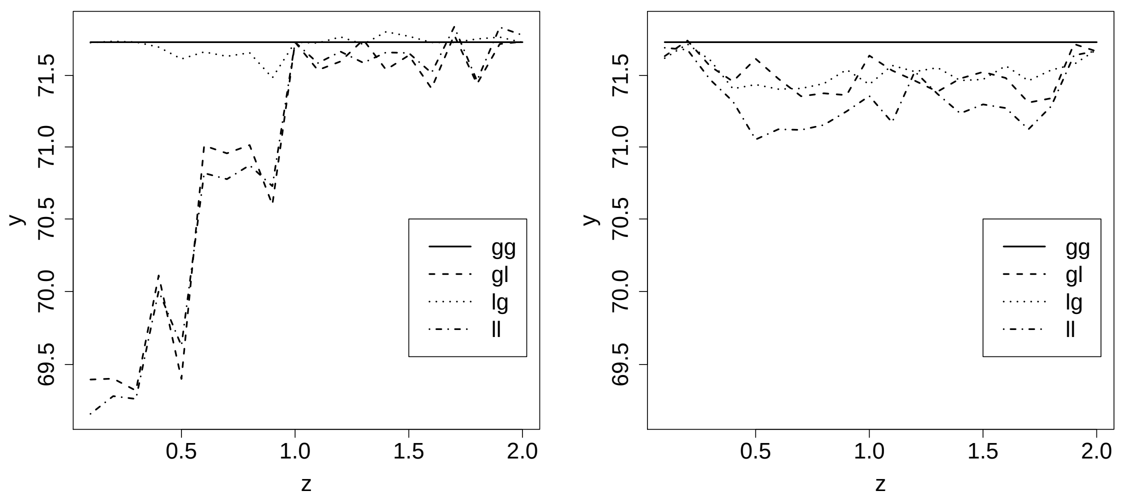

Figure 4: Function values plotted against z for the different modeling approaches for exprg09, the wild function. *Left:* without moving, the gl and ll models show a performance increase for small z values. *Right:* with moving, all modeling approaches perform better than gg where no moving or zooming is applied.

Results from exprg09 show similar results, with the difference of the performance without moving, as shown in Fig. 4 (left). The herein optimized wild function has many local optima. Hence, small values of z, where shrinking is applied if no better solution is found, seems to be beneficial for the local prediction modeling approaches gl and ll, but not with local meta modeling. This can be explained by understanding how the modeling is applied. A strongly localized meta model is, due to the small sample sizes, not able to model the overall behavior of the wild function. At the same time, a localized prediction on a global model leads to a good search direction.

Fig. 5 displays results from expgr08 (left) and expgr10 (right), both with moving. Exprg10 shows a performance similar to expgr07. For expgr08, the z range was extended to see the behavior for large z values. Exprg08 has a difficult large plateau on the left side, where optimization algorithms usually fail to improve (and perform similar to a random search). Without moving (not shown) no benefit can be achieved with local modeling approaches. With moving, the optimum is often found for z values between 2 and 4, which imply a large shrinking if a good solution is found, together with local prediction models.

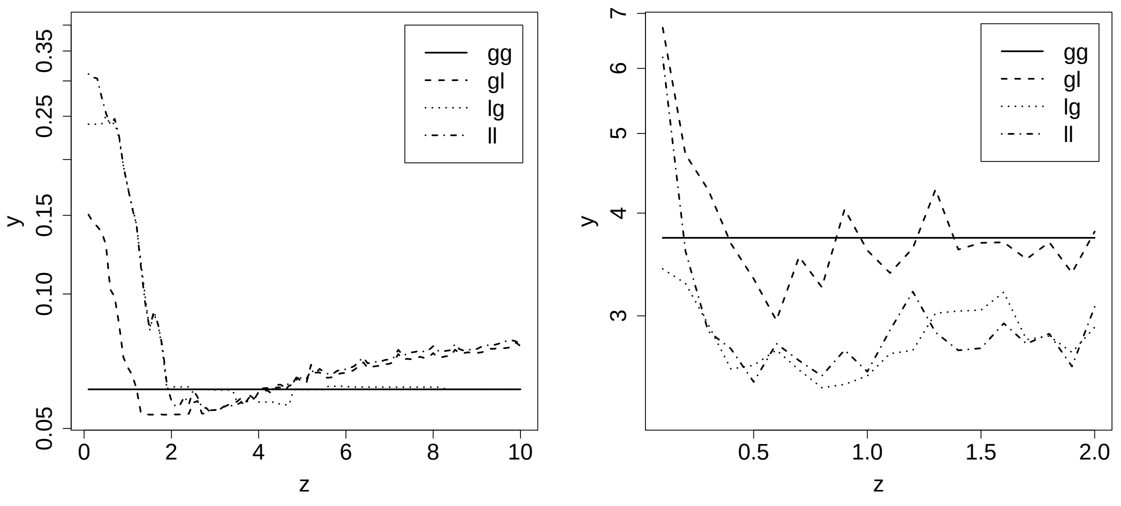

Figure 5: Function values plotted against z for the different modeling approaches. *Left:* Results from exprg08 with extended range of z, with moving. *Right:* Results from exprg10, with moving.

6 Conclusion

Regarding our main research question, how *moving* and *zooming* with different local model building affect the performance, the following insights were obtained:

- With *moving* and *zooming* a performance increase can be obtained
- The correct setting of the parameter z is important
- The four different model training/exploration approaches strongly affect the performance of the algorithm, depending on the underlying problem.

Our results have shown, that for each of our test problems there is at least one model building variant which outperforms the default without zooming or moving. The variant with lg with local meta modeling and a global prediction modeling with active moving seems to have the best generalization ability, as the performance is at least steady or improved on all test problems. For some problems, we could observe some interesting artifacts, which also might point in the direction of further improvement for our algorithm: if the ROI is centered around the optimum, zooming in without moving has a significant positive effect. We anticipated this, but it shows that the general idea is working and there is a potential for significant improvement. An also interesting effect takes place on the more complex wild function, where a localized search in case of no improvement of the found solution can lead to a significant boost in performance. So we still need to have a good problem knowledge for choosing the correct setting of z and the best model.

This study provides some interesting leads for future work on the topic of adaptive ROIs. We need to revise the moving and shrinking algorithm and try to find an adaptive solution for choosing z and the correct model building. The adaptive process is intended to improve the performance without interaction of the user. Further experiments should be able to give us more insight. We particularly need to perform experiments on higher dimensional cases as most tuning problems consist of a large set of parameters. Further, in future the algorithm shall be able to exceed the user predefined limits (to a still user-set absolute hard limit) so it is possible to enlarge the search space while keeping a initial local approach.

References

[1] Eiben, A. E.; Smith, J. E.: *Introduction to Evolutionary Computing*. Berlin, Heidelberg, New York: Springer. 2003.

[2] Bartz-Beielstein, T.; Branke, J.; Mehnen, J.; Mersmann, O.: Evolutionary Algorithms. *WIREs Data Mining and Knowledge Discovery* 4 (2014), S. 178–195.

[3] Stützle, T.; López-Ibáñez, M.: Automatic (Offline) Configuration of Algorithms. In: *GECCO '14: Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*. New York, NY, USA: ACM. 2014.

[4] Beyer, H.-G.; Schwefel, H.-P.: Evolution Strategies: A Comprehensive Introduction. *Natural Computing* 1 (2002) 1, S. 3–52.

[5] Box, G. E. P.; Draper, N. R.: *Empirical Model Building and Response Surfaces*. New York NY: Wiley. 1987.

[6] Nannen, V.; Eiben, A. E.: Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters. In: *Proc of IJCAI 2007* (Veloso, M., Hg.), S. 975–980. AAAI Press. 2007.

[7] Nannen, V.; Eiben, A. E.: Efficient Relevance Estimation and Value Calibration of Evolutionary Algorithm Parameters. In: *IEEE Congress on Evolutionary Computation*, S. 103–110. Singapore: IEEE Press, Piscataway, NJ. 2007.

[8] Nannen, V.; Smit, S. K.; Eiben, A. E.: Costs and Benefits of Tuning Parameters of Evolutionary Algorithms. In: *PPSN* (Rudolph *et al*, G., Hg.), Bd. 5199 von *Lecture Notes in Computer Science*, S. 528–538. Springer. ISBN 978-3-540-87699-1. 2008.

[9] Smit, S. K.; Eiben, A. E.: Beating the 'world champion' evolutionary algorithm via REVAC tuning. In: *IEEE Congress on Evolutionary Computation*, S. 1–8. IEEE Computational Intelligence Society, Barcelona, Spain: IEEE Press. URL `http://www.cs.vu.nl/~sksmit/bibs/CEC-2010-RobustParameters.pdf`. 2010.

[10] Bartz-Beielstein, T.; Lasarczyk, C.; Preuß, M.: Sequential Parameter Optimization. In: *Proceedings 2005 Congress on Evolutionary Computation (CEC'05), Edinburgh, Scotland* (McKay, B.; et al., Hg.), Bd. 1, S. 773–780. Piscataway NJ: IEEE Press. 2005.

[11] Bartz-Beielstein, T.: Sequential Parameter Optimization—An Annotated Bibliography. CIOP Technical Report 04/10, Research Center CIOP (Computational Intelligence, Optimization and Data Mining), Cologne University of Applied Science, Faculty of Computer Science and Engineering Science. 2010.