SACOBRA: Self-Adjusting Constrained Black-Box Optimization with RBF

Samineh Bagheri*, Wolfgang Konen*, Christophe Foussette⊕, Peter Krause⊕, Thomas Bäck△, Patrick Koch

*TH Köln (University of Applied Sciences)
Steinmüllerallee 1
51643 Gummersbach
E-Mail: {wolfgang.konen, samineh.bagheri}@th-koeln.de

⊕divis intelligent solutions GmbH
Joseph-von-Fraunhofer-Str. 20
44227 Dortmund
E-Mail: {foussette, krause, baeck}@divis-gmbh.de

△Leiden University, LIACS
2333 CA Leiden, The Netherlands
E-Mail: T.H.W.Baeck@liacs.leidenuniv.nl

Abstract

Modern real-world optimization problems are often high dimensional and subject to many constraints. These problems are typically expensive in terms of cost and computational time. Conventional constraint-based solvers often require a high number of function evaluations which are not affordable for such problems in practice. Employment of fast surrogate models to approximate objective and constraint functions is a known approach for efficient optimization. In this paper we present a new algorithm called **Self-Adjusting COBRA (SACOBRA)** based on Regis' COBRA [1]. We evaluate our approach by using 11 G-problems. We get very good results on 10 of the 11 G-problems with a severely limited budget of only 300 evaluations.

Introduction

Constrained optimization is often much harder than unconstrained optimization since the multitude of functions (one objective, numerous constraints) may constitute highly conflicting goals. If these functions are in addition expensive to evaluate, most algorithms make use of surrogate models.

The strength of surrogate-assisted techniques relies on the correct choice of modeling technique. In contrast to many surrogate modeling techniques like Kriging and support vector machines [2, 3], the performance of **radial basis function (RBF)** is only weakly dependent on the dimensionality of the optimization tasks or number of design points [4]. Therefore, in the area of efficient surrogate-assisted optimization a lot of attention is devoted to RBF modeling [1, 5, 6, 7]. COBRA [1] – an efficient constrained-based optimizer which uses RBF surrogates – outperforms many other algorithms on a large number of benchmarks. COBRA-R [6, 7] is a variant of COBRA implemented in `R` [8] with extended initialization methods and a novel repair technique for infeasible solutions [7].

Although both extensions gained remarkable success, the achievement was reached only after careful selection of the parameters and preprocessing transformations for each problem [6]. In the case of black-box optimization manual parameter tuning is not a valid approach since it requires some information about each single problem. Furthermore, automatic exploration of the parameter space is expensive. Therefore, some recent studies focused on automatic parameter tuning based on some measurable features of the problem [9].

In this extended abstract, we outline the goal to develop *one* algorithm which decides on its own about the best parameter setting and transformation functions. The new algorithm, which we call **Self-Adjusting COBRA (SACOBRA)**, is able to adjust those parameters automatically after the initialization phase. Additionally, SACOBRA incorporates a **random restart** algorithm to avoid occasional early stagnation and bad solutions due to an unfortunate initial design.

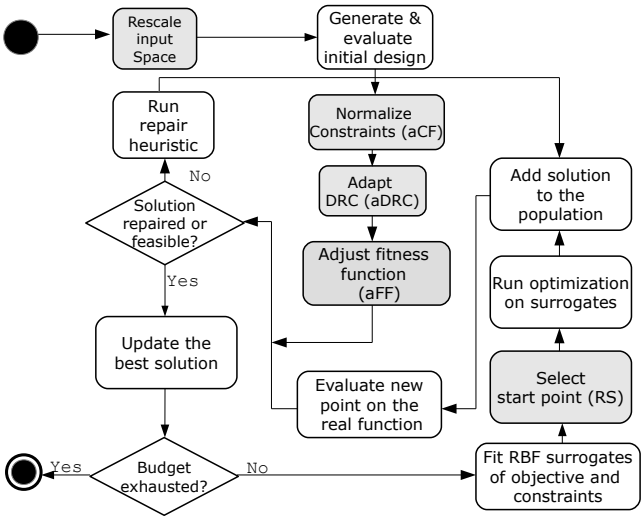

Figure 1: SAOCBRA algorithm proposed in this paper. The SACOBRA extensions are the grey boxes. Using only the white boxes results in the former COBRA-R algorithm.

Methods

Constrained optimization

A constrained optimization problem can be defined by the minimization of an objective function f subject to constraint function(s) $g_1, \ldots, g_m$:

$$\begin{aligned} &\text{Minimize} && f(\vec{x}), \vec{x} \in \mathbb{R}^d \\ &\text{subject to} && \\ & && g_i(\vec{x}) \leq 0, i = 1, 2, \ldots, m \end{aligned}$$

In this paper we always consider minimization problems. Maximization problems are transformed to minimization without loss of generality.

COBRA: Constrained Optimization by Radial Basis Function Approximation

Constrained Optimization by Radial Basis Function Approximation (COBRA) (Figure 1, only white boxes) is an efficient optimization algorithm proposed by Regis [1]. This method relies on modeling the objective function and constraint function(s) by means of radial basis functions (surrogates) in order to minimize evaluations of the real objective and constraint functions. Each iterate is a result of an optimization on the surrogates. Only at the end of each iteration the real functions are evaluated once and a new solution is added to the population. COBRA-R [6] is a re-implementation of COBRA [1] in R [8] with some extensions which are described in detail in [6].

An important element of COBRA and COBRA-R is the *Distance Requirement Cycle (DRC)*: In order to facilitate exploration, Regis [1] proposes to add in each iteration i additional constraints enforcing that the current solution $\vec{x}$ is at least a distance ρ_i away from all previous iterates. The distance ρ_i is taken cyclically from a small set of distances, the so-called DRC. Large ρ_i enforce exploration, small ρ_i exploitation. It turns out that different problems are quite sensitive to the right choice of DRC.

Repair algorithm RI-2

Sometimes the infill points returned by the internal optimizer are infeasible. A repair algorithm is embedded in the COBRA-R optimization framework which intends to repair infill points with a slight infeasibility by guiding them to the feasible region. The repair algorithm RI-2 used in COBRA-R is described and discussed in detail in [7]. It is worthwhile to mention that the repair algorithm is performed on the surrogate models, so no real function evaluations are necessary for this repair.

SACOBRA: Self-Adjusting COBRA

SACOBRA, the self-adjusting COBRA-R algorithm, includes five extensions (Figure 1, with grey boxes) in comparison to the COBRA-R optimization framework [7]:

- **Rescale the input space:** The input vector $\vec{x}$ is element-wise rescaled to $[-1, +1]$. This helps to have a better exploration all over the search space because all dimensions are treated the same and also avoids numerical instabilities caused by high values of $\vec{x}$.
- **Random Start algorithm (RS)**: Normally COBRA starts optimization from the best point found so far. With RS the optimization starts with a certain probability from a random point in the search space. RS is especially beneficial when the search gets stuck in local optima.
- **Adjusting DRC parameter (aDRC)** is done after the initialization phase. Our experimental analysis showed that large DRC values set can be harmful for problems with a very steep objective function, while those large values are needed for other problems. Therefore, we developed an automatic DRC adjustment which selects the appropriate DRC set according to the information extracted after the initialization phase.
- **Adjusting fitness function (aFF)** is developed to modify objective functions which are challenging to model with RBF. Our experimental analysis revealed that RBF often do not provide a proper model for functions which are very steep. A logarithmic transformation of the objective function can be helpful. SACOBRA internally decides whether a logarithmic transformation of the objective function is necessary.
- **Adjusting constraint function(s) (aCF)** is actually done by normalizing the range of constraint functions for each problem. Embedding this step boosts up the optimization performance because all constraints have values in a similar range.

The details of the internal adjustment algorithms and the role of problem-specific features like FR and GR (see Tab. 1 below) will be presented in a forthcoming publication.

Experimental Setup

We evaluate our proposed technique by using a well-studied test suite of G-problems described in [10]. The diversity of the G-problem characteristics makes them a very challenging benchmark for optimization techniques. In Table 1 we show features of these functions. In this study we identified two new features FR and GR (defined in Table 1) which constitute useful elements for our self-adjusting procedures.

We apply the SACOBRA algorithm to 330 problems: G01-G11, initialized with 30 different initial populations. The initial population of size $3 \cdot d$ is generated by means of Latin hypercube sampling. The optimization on surrogates is done by COBYLA [11]. Cubic RBFs are utilized to model the functions. The DRC parameters are selected internally among $DRCL = \langle 0.3, 0.05, 0.001, 0.0005, 0.0 \rangle$ and $DRCS = \langle 0.001, 0.0 \rangle$. In order to measure the strength of each extension we repeat all tests in absence of each extension. The importance of each extension is measured with the Wilcoxon signed rank sum test of the final optimization error (Table 2), which is a one sided, paired test on the 30 pairs with different seeds.

Results and Discussion

In our previous work [6] we established good results with COBRA-R on most G-problems, but we had to manually tune algorithmic configurations and parameters for each G-problem anew, which is a tedious and time-consuming procedure. Now we are in a position to present with SACOBRA an algorithm which runs with the same settings on all G-problems and produces good results with relatively few function evaluations (100–300). We will show in the talk and in a forthcoming publication the performance profiles [12] for SACOBRA and its variants, leading to a substantial improvement over COBRA-R with *fixed* parameter settings. SACOBRA solves 85% of the 330 test problems while COBRA-R solves only 60%.

We can identify with our results the elements of the SACOBRA algorithm with greatest impact: Most important are **rescaling** and **random restart**, followed by **automatic fitness function adjustment**. Least important are **aDRC** and **aCF**.

Table 1: Characteristics of the G-functions: d: dimension, ρ^*: feasibility rate (%) after changing equality constraints to inequality constraints, FR: range of the fitness values, GR: ratio of largest to smallest constraint range, LI: number of linear inequalities, NI: number of nonlinear inequalities, NE: number of nonlinear equalities, a: number of constraints active at the optimum.

Fct.	d	type	ρ^*	FR	GR	LI	NI	NE	a
G01	13	quadratic	0.0003%	298.14	1.969	9	0	0	6
G02	10	nonlinear	99.997%	0.57	2.632	1	1	0	1
G03	20	nonlinear	0.0000%	92684985979.23	1.000	0	0	1	1
G04	5	quadratic	26.9217%	9832.45	2.161	0	6	0	2
G05	4	nonlinear	0.0919%	8863.69	1788.74	2	0	3	3
G06	2	nonlinear	0.0072%	1246828.23	1.010	0	2	0	2
G07	10	quadratic	0.0000%	5928.19	12.671	3	5	0	6
G08	2	nonlinear	0.8751%	1821.61	2.393	0	2	0	0
G09	7	nonlinear	0.5207%	10013016.18	25.05	0	4	0	2
G10	8	linear	0.0008%	27610.89	3842702	3	3	0	3
G11	2	linear	66.7240%	4.99	1.000	0	0	1	1

Tab. 2 shows that each of these elements has its importance (is relevant) for some of the G-problems: M1 is significantly better than M$*$ at least for some G-problems. And each G-problem benefits from one or more SACOBRA extensions. The only exceptions from this rule are G02-10d and G11, but for different reasons: G02-10d is a high-dimensional and highly multimodal problem which is generally hard to solve by *any* of the SACOBRA- or COBRA-variants (the reason is that surrogate models with a low number of points cannot capture enough detail of this complicated fitness function). G11 on the other hand is an easy problem which is solved by *all* SACOBRA variants, so none is significantly better than the others.

Conclusion

We presented with SACOBRA a self-adjusting algorithm for expensive constrained optimization which can successfully solve a variety of challenging benchmark problems without any problem-specific parameter tuning. It

Table 2: Wilcoxon rank sum test, paired, one sided, significance level 5%. Shown is the p-value. Significant tests ($p < 5\%$) are marked as gray cells. Optimization methods: M1: SACOBRA, M2: SACOBRA\rescale (SACOBRA without rescaling the input space), M3: SACOBRA\RS (SACOBRA without random start), M4: SACOBRA\aDRC, M5: SACOBRA\aFF, M6: SACOBRA\aCF, M7: COBRA ($Xi = DRCS$), M8: COBRA ($Xi = DRCL$).

func\method	M1M2	M1M3	M1M4	M1M5	M1M6	M1M7	M1M8
G01	0.177	4.8e-02	0.887	1.000	1.000	0.167	4.8e-02
G02-10d	0.144	0.664	0.743	0.789	0.757	0.074	0.985
G03	1.000	8.7e-03	1.000	1.4e-06	1.000	2.4e-05	1.4e-06
G04	0.230	0.121	1.000	0.909	0.909	0.121	9.1e-07
G05	1.4e-03	0.557	1.000	0.676	0.138	0.999	0.997
G06	9.1e-07	0.745	1.000	0.230	0.500	0.194	9.1e-07
G07	0.745	0.516	1.000	0.352	0.352	0.516	2.0e-06
G08	7.6e-03	0.112	3.7e-03	1.000	1.000	5.4e-04	0.112
G09	0.971	0.594	1.000	1.5e-05	0.884	1.5e-05	1.4e-06
G10	3.1e-03	2.7e-03	1.000	0.843	2.5e-02	1.1e-03	2.0e-03
G11	0.660	1.000	0.875	1.000	1.000	1.000	1.000

does so by self-adjusting its internal parameters to the characteristics of the problem, either after the initialization phase or online during iterations.

SACOBRA solves 10 of the 11 G-problems. The only exception is G02-10d, which is a highly multimodal problem. Such problems cannot be solved in a few iterations with the current surrogate models. The investigation of surrogate modeling for highly multimodal functions is a topic of our future research.

Acknowledgements

This work has been supported by the Bundesministerium für Wirtschaft (BMWi) under the ZIM grant MONREP (AiF FKZ KF3145102, Zentrales Innovationsprogramm Mittelstand).

Supported by:

Federal Ministry for Economic Affairs and Energy

on the basis of a decision by the German Bundestag

Literature

[1] Regis, R.: Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. *Engineering Optimization* 46 (2013) 2, S. 218–243.

[2] Poloczek, J.; Kramer, O.: Local SVM constraint surrogate models for self-adaptive evolution strategies. In: *Proceedings of the 36th Annual German Conference on Advances in Artificial Intelligence (KI 2013)*, S. 164–175. Springer. 2013.

[3] Emmerich, M.; Giotis, A.; Özdemir, M.; Bäck, T.; Giannakoglou, K.: Metamodel-Assisted Evolution Strategies. In: *Proceedings of the Conference on Parallel Problem Solving from Nature (PPSN VII)* (Guervós, J. M.; et al., Hg.), Bd. 2439 von *Lecture Notes in Computer Science*, S. 361–370. Springer Berlin Heidelberg. 2002.

[4] Buhmann, M. D.: *Radial Basis Functions*. New York, NY, USA: Cambridge University Press. ISBN 0521633389. 2003.

[5] Sun, C.; Jin, Y.; Zeng, J.; Yu, Y.: A two-layer surrogate-assisted particle swarm optimization algorithm. *Soft Computing* 19 (2015) 6, S. 1461–1475.

[6] Koch, P.; Bagheri, S.; Foussette, C.; Krause, P.; Bäck, T.; Konen, W.: Constrained Optimization with a Limited Number of Function Evaluations. In: *Proceedings of the 24th Workshop on Computational Intelligence* (Hoffmann, F.; Hüllermeier, E., Hg.), S. 119–134. Universitätsverlag Karlsruhe. Young Author Award GMA-CI. 2014.

[7] Koch, P.; Bagheri, S.; Konen, W.; Foussette, C.; Krause, P.; Bäck, T.: A New Repair Method For Constrained Optimization. In: *Proceedings of the Genetic and Evolutionary Computation Conference 2015 (GECCO)* (Jiménez-Laredo, J. L., Hg.), S. 273–280. ACM. 2015.

[8] R Core Team: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL `http://www.R-project.org`. 2014.

[9] Bossek, J.; Bischl, B.; Wagner, T.; Rudolph, G.: Learning Feature-Parameter Mappings for Parameter Tuning via the Profile Expected Improvement. In: *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference*, GECCO '15, S. 1319–1326. New York, NY, USA: ACM. ISBN 978-1-4503-3472-3. 2015.

[10] Michalewicz, Z.; Schoenauer, M.: Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation* 4 (1996) 1, S. 1–32.

[11] Powell, M.: A direct search optimization method that models the objective and constraint functions by linear interpolation. In: *Advances In Optimization And Numerical Analysis*, S. 51–67. Springer. 1994.

[12] Moré, J. J.; Wild, S. M.: Benchmarking derivative-free optimization algorithms. *SIAM J. Optimization* 20 (2009) 1, S. 172–191.