

# Surrogate Constraint Functions for CMA Evolution Strategies

Oliver Kramer, André Barthelmes, and Günter Rudolph

Department of Computer Science  
Technische Universität Dortmund  
44227 Dortmund, Germany

**Abstract.** Many practical optimization problems are constrained black boxes. Covariance Matrix Adaptation Evolution Strategies (CMA-ES) belong to the most successful black box optimization methods. Up to now no sophisticated constraint handling method for Covariance Matrix Adaptation optimizers has been proposed. In our novel approach we learn a meta-model of the constraint function and use this surrogate model to adapt the covariance matrix during the search at the vicinity of the constraint boundary. The meta-model can be used for various purposes, i.e. rotation of the mutation ellipsoid, checking the feasibility of candidate solutions or repairing infeasible mutations by projecting them onto the constraint surrogate function. Experimental results show the potentials of the proposed approach.

## 1 Introduction

Constraints typically are not considered available in their explicit formal form, but are assumed to be black boxes: a vector  $\mathbf{x}$  fed to the black box just returns a numerical or boolean value. The constrained real-valued optimization problem is to find a solution  $\mathbf{x} = (x_1, x_2, \dots, x_N)^T$  in the  $N$ -dimensional solution space  $\mathbb{R}^N$  that minimizes the objective function  $f(\mathbf{x})$ , in symbols:

$$\begin{aligned} f(\mathbf{x}) &\rightarrow \min!, \mathbf{x} \in \mathbb{R}^N && \text{subject to} \\ \text{inequalities} & g_i(\mathbf{x}) \leq 0, i = 1, \dots, n_1, \text{ and} && (1) \\ \text{equalities} & h_j(\mathbf{x}) = 0, j = 1, \dots, n_2. \end{aligned}$$

The question arises how to cope with constraints that are not given explicitly. Various constraint handling methods for evolutionary computation have been proposed in the last decades. A survey is not possible due to the limited space. Coello [1] or Kramer [4] deliver a good starting point for literature research. Most methods fall into the category of penalty functions, e.g. by Kuri *et al.* [5] allowing the search process to discover the whole search space, but penalizing the infeasible part. Other approaches are based on decoders, e.g. by Michalewicz [7] or multi-objective optimization, e.g. by Jimenez [3].

Covariance Matrix Adaptation belongs to the most successful black box optimization algorithms in real-valued search spaces. The idea of covariance matrix

adaptation techniques is to adapt the distribution of the mutation operator such that the probability to reproduce steps that led to the actual population increases. The CMA-ES was introduced by Ostermeier et al. [8,2]. All previous CMA-ES implementations make use of death penalty as constraint handling technique. The results of the CMA-ES on problems TR2 and 2.40<sup>1</sup> can be found in table 1. In comparison to a standard evolution strategy the CMA-ES is able to cope with the low success rates around the optimum of the TR-problem. The average number of infeasible solutions during the approximation is 44%. This indicates that a reasonable adaptation of the mutation ellipsoid takes place. As the TR problem exhibits a linear objective and a linear constraint function we would expect a rate of  $\approx 50\%$  infeasible solutions and a decrease of step sizes for an ill-adapted mutation ellipsoid. An analysis of the angle between the main axis of the mutation ellipsoid and the constraint function converges to zero, the same do the step sizes during approximation of the optimum. On problem 2.40 about 64% of the produced solutions are infeasible.

**Table 1.** Experimental analysis of the CMA-ES with death penalty

CMA-ES (DP)	best	mean	worst	dev	ffe	cfe
TR2	0.0	0.0	$2.7 \cdot 10^{-15}$	$5.8 \cdot 10^{-16}$	6,754	12,019
2.40	0.0	0.0	$9.1 \cdot 10^{-13}$	$1.3 \cdot 10^{-13}$	19,019	71,241

## 2 A Fast Surrogate Model Estimator for Linear Constraints

To construct more powerful constraint handling methods the idea arises to build a meta-model of the constraint function. This meta-model can be used in various kinds of ways. We will use the surrogate model for the rotation of the mutation ellipsoid, checking the feasibility of mutations, and the repair of infeasible mutations of the CMA-ES. In this work we concentrate on linear constraints, but the approach can be extended to the non-linear case.

To efficiently estimate the constraint surrogate function we developed a new efficient constraint boundary estimator based on fast binary search. We assume that our constraint handling method starts with the first occurrence of an infeasible individual. The estimation algorithm for the linear constraint boundary, i.e. a linear hyper-plane  $h_0$  works as follows.

In the first step the surrogate estimator identifies  $N$  points *on* the  $N$ -dimensional linear constraint hyper-plane  $h_0$ . For this sake the model-estimator makes use of an  $N$ -dimensional hypersphere cutting the constraint boundary. The connection between the infeasible points on the hypersphere and the center of the hypersphere will cut the constraint boundary. When the first infeasible offspring individual  $(\xi)_1$  is produced, the original feasible parent  $\mathbf{x}_f$  is the center of the

<sup>1</sup> Problem TR2 is the Sphere function  $\mathbf{x}^T \mathbf{x}$  with the linear constraint  $\sum x_i - N > 0$ , problem 2.40 is a linear function with 6 linear constraints, see Schwefel [9].

corresponding meta-model estimator and the distance becomes radius  $r$  of your model-estimator. The approach uses binary search to find the cutting points and takes the following three steps:

1. Generation of random points on the surface of the hypersphere: Point  $\mathbf{x}_f$  is the center of the hypersphere with radius  $r$  such that the constraint boundary is cut.  $N - 1$  additional infeasible points  $\xi_i$ ,  $1 \leq i \leq N - 1$  have to be produced. Infeasible points are produced by sampling on its surface until a sufficient number of infeasible points are produced. The points on the surface are sampled randomly with uniform distribution using the method of Marsaglia [6]: At first,  $N - 1$  Gaussian distributed points are sampled  $(\xi_j)_i \sim \mathcal{N}(0, 1)$   $j = 1, \dots, N$  and scaled to length 1. Further scaling and shifting yields  $N$  randomly distributed point on the hypersphere surface

$$(\xi)_i = \frac{1}{\sqrt{(\xi_1)_i^2 + \dots + (\xi_N)_i^2}} \cdot ((\xi_1)_i, \dots, (\xi_N)_i)^T \quad (2)$$

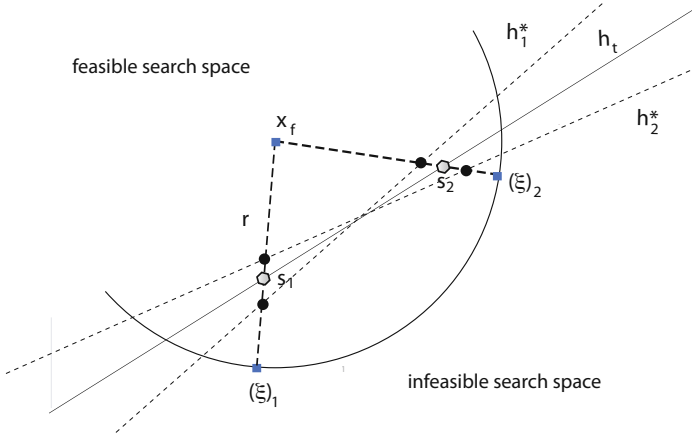
2. Identification of  $N$  points  $s_1, \dots, s_N$  on the estimated constraint boundary, see Figure 1: The line between the feasible point  $\mathbf{x}_f$  and the  $i$ -th infeasible point  $(\xi)_i$  cuts the real constraint hyper-plane  $h^*$  in point  $s_i^*$ . Binary search is a fast approach to approximate  $s_i^*$  on this segment: The center  $s_i$  of the last interval defined by the last points of the binary search becomes an estimation of point  $s_i^*$  on  $h^*$ . The real hyperplane lies between  $h_1^*$  and  $h_2^*$ . Table 2 shows the empirical number of binary search steps to achieve the angle accuracy  $\phi < 1^\circ$  with accuracy  $\phi < 0.25^\circ$  of the estimated hyperplane  $h_0$ . Each experiment was repeated 100,000 times, i.e. in each repetition a new test case with a random hyper-plane and random points  $\mathbf{x}_f, (\xi)_1, \dots, (\xi)_N \in \mathbb{R}$  was generated. The error of the angle  $\phi$  can be estimated by the number of binary search steps: Let  $\phi_k$  be the average angle error after  $k$  binary steps and  $\eta$  be the accuracy improvement factor we are able to achieve with one binary search step. The experimental analysis yields the following linear relation between angle accuracy  $\phi$  and binary search steps  $k = j - i$ :

$$\phi_i \cdot \eta^{j-i} = \phi_j \quad (3)$$

The magnitude of the efficiency factor  $\eta$ , i.e. an improvement of angle accuracy, can be estimated as  $0.53 \leq \eta \leq 0.57$  from our experiments, see Table 2. Note, that the number of binary search steps grows slower than the number of dimensions of the solution space.

3. Finally, we calculate the normal vector  $\mathbf{n}_0$  of  $h_0$  using the  $N$  points on the constraint boundary. Assuming the points  $s_i$ ,  $1 \leq i \leq N$ , represent linearly independent vectors a successive Gram-Schmidt orthogonalization of the  $(i + 1)$ -th vector on the  $i$ -th previously produced vectors delivers the normal vector  $\mathbf{n}_0$  of  $h_0$ .

Note, that we estimate the normal vector  $\mathbf{n}_0$  of the linear constraint model  $h_0$  only one time, i.e. when the first infeasible solutions have been detected. Later,



**Fig. 1.** Procedure to estimate the constraint boundary  $h_0$  in two dimensions: The method performs binary search on the segments between a feasible point  $x_f$  and each two infeasible points  $(\xi)_1, (\xi)_2$  to estimate two points  $s_1, s_2$  on the meta-model

**Table 2.** Number of binary search steps and constraint function evaluations to limit the average angle accuracy in an artificial setup with one hyperplane, a feasible and an infeasible point

dimension	steps ( $\phi \leq 1^\circ$ )	mean error	steps ( $\phi \leq 0.25^\circ$ )	mean error	cfe
2	9	0.85	11	0.24	3.14
10	14	0.68	16	0.21	21.56
20	15	0.76	17	0.25	42.96

update steps only concern the local support point  $p_t$  of the hyper-plane<sup>2</sup>. At the beginning, any of the points  $s_i$  may be the support point  $p_0$ .

### 3 Rotation of the Covariance Matrix

Our first idea to exploit the surrogate constraint function was to adapt the covariance matrix itself. Our experiments with correlated Gaussian mutation for ES motivated that step<sup>3</sup>. The covariance matrix  $C \in \mathbb{R}^{N \times N}$  is a positive semi-definite matrix and thus, exhibits a decomposition into an orthonormal matrix  $B$  and a diagonal matrix  $D$ , such that  $C = BD^2B^T$ . In case of approximating the constraint boundary  $h$  with normal vector  $n$  we adapt the covariance matrix  $C$  as follows:

<sup>2</sup> Hence, in iteration  $t$  the linear model  $h_t$  is specified by normal vector  $n_0$  and current support point  $p_t$ .

<sup>3</sup> Correlated mutation by Schwefel [9] rotates the axes of the hyper-ellipsoid to adapt to local properties of the fitness landscape by means of self-adaptation.

1. Decomposition of covariance matrix  $\mathbf{C}$  into orthonormal matrix  $\mathbf{B}$  and diagonal matrix  $\mathbf{D}$ .
2. Replacement of vector  $\mathbf{c}_i$  of matrix  $\mathbf{C}$  with the least angle  $d$  to the normal vector  $\mathbf{n}_0$ , i.e.  $\mathbf{c}_i$  such that  $\mathbf{c}_i \mathbf{n}_0$  is minimal, by normalized vector  $\mathbf{n}_0 / \|\mathbf{n}_0\|$ .
3. Orthogonalization of  $\mathbf{C}$  starting with  $\mathbf{n}_0 / \|\mathbf{n}_0\|$ .
4. Scale of the  $N - 1$  vectors of  $\mathbf{D}$  that are orthogonal to  $\mathbf{n}$  with regard to a projection of  $\mathbf{C}$  onto  $h$ , and scale of the vector parallel to  $\mathbf{n}$  to 1/200-th of its original length.
5. Computation of  $\mathbf{C} = \mathbf{B}\mathbf{D}^2\mathbf{B}^T$ .

The CMA-ES runs as usual until the constraint boundary is met. The algorithm starts the constraint handling technique if 40% of the mutations are infeasible. Figure 3 shows the results of our experimental analysis on TR2 and 2.40. We can observe a slight decrease of necessary fitness function calls of about 5% in average on TR2. This slight speedup is a result of a faster adaptation of  $\mathbf{C}$  to the constraint boundary. After the adaptation is complete, the optimization process continuous in the usual kind of way. No speedup can be observed on problem 2.40, the CMA-ES was already successful in adapting  $\mathbf{C}$  to the constraint conditions.

**Table 3.** Experimental analysis of the CMA-ES using the surrogate model for adaptation of  $\mathbf{C}$

CMA-ES (DP)	best	mean	dev	ffe	cfe
TR2	0.0	0.0	$3.1 \cdot 10^{-15}$	6,380	11,374
2.40	0.0	0.0	$1.3 \cdot 10^{-13}$	18,998	71,559

## 4 Checking Feasibility

In a second approach potentially feasible solutions are checked for feasibility with the surrogate model. In particular, this is recommendable if constraint function calls are expensive. For this purpose an exact estimation of the constraint boundary in necessary. Two errors for the feasibility prediction of individual  $\mathbf{x}_t$  are possible:

1. The model predicts  $\mathbf{x}_t$  is feasible, but it is not. Points of this category are examined for feasibility causing an additional constraint function evaluation.
2. The model predicts  $\mathbf{x}_t$  is infeasible, but it is feasible. The individual will be discarded, but may be a very good approximation of the optimum.

We introduce a *safety margin*  $\delta$ , i.e. a shift of the estimated constraint boundary into the infeasible direction. The safety margin reduces errors of type 2. We set  $\delta$  to the distance  $d$  of the mutation ellipsoid center  $c$  and the estimated constraint boundary  $h_t$ . The distance between  $c$  and the shifted constraint boundary  $h'_t$  becomes  $2d$ . A regular update of the constraint boundary support point  $p_t$  is

necessary. Again, we have to distinguish between two conditions in each iteration. Let  $d_{t_0}$  be the distance between the mutation ellipsoid center  $c_{t_0}$  and the constraint boundary  $h_{t_0}$  at time  $t_0$  and let  $k$  be the number of binary search steps to achieve the angle accuracy of  $\delta < 0.25^\circ$ .

1. The center of the mutation ellipsoid  $c_t$  approaches  $h_t$ : If distance  $d_t$  between  $h_t$  and  $c_{t_0}$  becomes smaller than  $d_{t_0}/2^k$ , a re-estimation of the support point  $p_t$  is reasonable.
2. The search  $c_t$  moves parallel to  $h_t$ : An exceeding of distance

$$c_{t_0} - c_t = \sqrt{\frac{1}{\tan(\phi)^2} + 4} \cdot d_{t_0} \quad (4)$$

with  $\phi = 0.25 \cdot (0.57)^{3k}$  causes a re-estimation of  $h_t$ .

To find the new support point  $p_t$  we use  $4k$  binary steps on the line between the current infeasible solutions and  $c_t$ . Table 4 shows significant savings of constraint evaluations with a high approximation capability using the feasibility checking approach.

**Table 4.** Results of the CMA-ES with meta-model feasibility checking

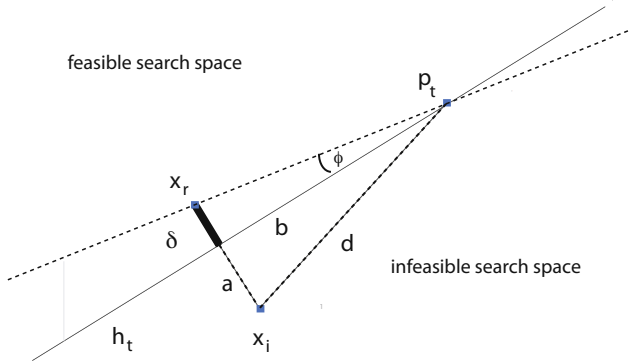
CMA-ES (F-Checking)	best	mean	dev	ffe	cfe
TR2	0.0	0.0	$6.9 \cdot 10^{-16}$	6,780	7,781
2.40	0.0	0.0	$1.8 \cdot 10^{-13}$	19,386	34,254

## 5 Repair of Infeasible Solutions

At last, we propose to repair infeasible mutations by projection onto the constraint boundary  $h_t$ . We assume the angle error  $\phi$  that can again be estimated by the number of binary search steps  $k$ , see equation 3. In our approach we elongate the projection vector by length  $\delta$ . Figure 2 illustrates the situation. Let  $p_t$  be the support point of the hyper-plane  $h_t$  at time  $t$  and let  $x_i$  be the infeasible solution. Since  $a^2 + b^2 = d^2$  and  $\delta/b = \tan \phi$ , we obtain

$$\delta = \sqrt{a^2 - d^2} \cdot \tan \phi. \quad (5)$$

The elongation of the projection into the potentially feasible region guarantees feasibility of the repaired individuals. Nevertheless, it might prevent fast convergence, in particular in regions far away from the hyper-plane support point  $p_t$ , as  $\delta$  grows with increasing length of  $d$ . In our approach we update the center of the hyper-plane for an update of accuracy every 10 generations. The results of the CMA-ES repair algorithm can be found in Table 5. We observe a significant decrease of fitness function evaluations. The search concentrates on the boundary of the infeasible search space, in particular on the feasible site. Of course, no saving of constraint function evaluations could have been expected.



**Fig. 2.** Safety margin: the elongation of the projection of infeasible solution  $x_i$  onto the estimated constraint boundary  $h_t$  by length  $\delta$  ensures the feasibility of the repaired point  $x_r$ .

**Table 5.** Results of the CMA-ES with repair mechanism

CMA-ES (Repair)	best	mean	dev	ffe	cfe
TR2	0.0	0.0	$5.5 \cdot 10^{-16}$	3,432	5,326
2.40	0.0	0.0	$9.1 \cdot 10^{-14}$	16,067	75,705

## 6 Combination of Approaches

At last, we combine all three introduced approaches. Table 6 shows the results of the combined constraint handling techniques on problems TR2 and 2.40. In comparison to the repair approach of Section 5 the number of constraint function calls decreases significantly and the relation between fitness and constraint function evaluations reaches the level of the feasibility check technique in Section 4. In comparison to the results with death penalty, see Table 1, a significant decrease of objective and constraint function evaluations can be observed.

**Table 6.** Results of the CMA-ES with covariance matrix rotation, feasibility check and repair mechanism

CMA-ES (all)	best	mean	worst	dev	ffe	cfe
TR2	0.0	0.0	$8.9 \cdot 10^{-16}$	$5.1 \cdot 10^{-16}$	3,249	3,650
2.40	0.0	0.0	$9.1 \cdot 10^{-13}$	$9.1 \cdot 10^{-14}$	11,216	30,069

## 7 Summary and Outlook

We have proposed the first sophisticated constraint handling technique for the CMA-ES. The better the constraints are known, the more information can be

investigated during the search process. Surrogate models of the constraint functions turn out to be very useful for handling of constraints. To compute a constraint surrogate function we proposed an efficient algorithm for the estimation of linear constraints based on sampling on the surface of a hypersphere and binary search on the lines into the infeasible part of the search space. Based on this model we used the surrogate constraint function for multiple purposes, i.e. rotation of the covariance matrix, checking of feasibility and repair of infeasible points. Significant savings of fitness and constraint evaluations can be observed. If the effort spent on the surrogate constraint model and the various techniques is profitable, depends on the application scenario.

As many constraints in practical applications are not linear, we plan to extend our approach to non-linear constraints. First experiments show that a linear surrogate approximation of non-linear constraints is sufficient and leads to amazingly good results. Furthermore, the approach can be extended concerning the handling of multiple constraints functions. For this purpose, it seems to be reasonable to develop a detailed case sensitive surrogate model.

## References

1. Coello Coello, C.A.: Theoretical and numerical constraint handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering* 191(11-12), 1245–1287 (2002)
2. Hansen, N.: The CMA evolution strategy: A tutorial. Technical report, TU Berlin, ETH Zürich (2005)
3. Jimenez, F., Verdegay, J.L.: Evolutionary techniques for constrained optimization problems. In: Zimmermann, H.-J. (ed.) 7th European Congress on Intelligent Techniques and Soft Computing (EUFIT 1999). Verlag Mainz, Aachen (1999)
4. Kramer, O.: *Self-Adaptive Heuristics for Evolutionary Computation*. Springer, Berlin (2008)
5. Kuri-Morales, A., Quezada, C.V.: A universal eclectic genetic algorithm for constrained optimization. In: *Proceedings 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT 1998)*, September 1998, pp. 518–522. Verlag Mainz, Aachen (1998)
6. Marsaglia, G.: Choosing a point from the surface of a sphere. *The Annals of Mathematical Statistics* 43, 645–646 (1972)
7. Michalewicz, Z., Fogel, D.B.: *How to Solve It: Modern Heuristics*. Springer, Berlin (2000)
8. Ostermeier, A., Gawelczyk, A., Hansen, N.: A derandomized approach to self adaptation of evolution strategies. *Evolutionary Computation* 2(4), 369–380 (1994)
9. Schwefel, H.-P.: *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology. Wiley Interscience, New York (1995)