Boundary Operators for Constrained Parameter Optimization Problems

Marc Schoenauer
CMAP – URA CNRS 756
Ecole Polytechnique
Palaiseau 91128, France
marc.schoenauer@polytechnique.fr

Zbigniew Michalewicz*
Department of Computer Science
University of North Carolina
Charlotte, NC 28223
zbyszek@uncc.edu

Abstract

In this paper we continue our earlier study [SM96] on boundary operators for constrained parameter optimization problems. The significance of this line of research is based on the observation that usually the global solution for many optimization problems lies on the boundary of the feasible region. Thus, for many constrained numerical optimization problems it might be beneficial to search just the boundary of the search space defined by a set of constraints (some other algorithm might be used for searching the interior of the search space, if activity of a constraint is not certain). We present a few boundary operators for a sphere and provide their experimental evaluation.

1 INTRODUCTION

Evolutionary techniques (whether genetic algorithms, evolution strategies, or evolutionary programming) usually have difficulties in solving constrained numerical optimization problems (for a recent survey on constraint-handling techniques for parameter optimization problems, see [MS96]). Various search operators have been proposed and investigated; several different constraint handling techniques have been experimented with. However, for many test cases, the results of experiments were far from being satisfactory. As already stated in our recent paper [SM96], one of the main reasons behind this failure is the inability of evolutionary systems to precisely search the boundary area between feasible and infeasible regions of the search space; in the case of optimization problems with active constraints, such ability is essential.

It seems that the evolutionary computation techniques have a huge potential in incorporating operators which search the boundary of feasible and infeasible regions in an efficient way. In this paper we discuss some possible boundary operators for such a search and illustrate this approach on a few test cases.

The paper is organized as follows. Section 2 introduces the problem, whereas section 3 presents the proposed approach for numerical constrained optimization based on the idea of searching only the boundary of the feasible search space. This approach is discussed in detail in section 4 where several boundary operators on a sphere are defined. These operators are experimented with on a few test cases; the test cases and the results of experiments are reported in section 5. Section 6 concludes the paper.

2 THE PROBLEM

Let us consider the following constrained numerical optimization problem:

Find $x \in \mathcal{S} \subset R^n$ such that

$$\begin{cases} f(x) & = & \min\{f(y);\ y \in \mathcal{S}\}, \\ g_i(x) & \leq & 0, \quad for \quad i = 1, \ldots, q, \\ g_i(x) & = & 0, \quad for \quad i = q+1, \ldots, m, \end{cases} \qquad (1)$$

where f and g_i are real-valued functions on $\mathcal{S}$. The set of feasible points (i.e., points satisfying the constraints (2) and (3)) is denoted $\mathcal{F}$. Restricting the search to the feasible region seems an elegant way to treat constrained problems: in [MA94], the algorithm maintains feasibility of all linear constraints using a set of closed operators, which convert a feasible solution (feasible in terms of linear constraints only) into another feasible solution.

* also at the Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237 Warsaw, Poland.

However, for nonlinear constraints this ideal situation is generally out of reach, and during the last few years, several methods have been proposed for handling such constraints by evolutionary algorithms. For an experimental comparison of some of these methods, see [MS96]). In this paper we return to the idea of maintaining feasibility of solution by a set of closed operators; however, the differences are that (1) constraints are nonlinear, and (2) the closed operators transform point(s) from the boundary of the feasible region into other points from this boundary. We discuss it fully in the next section of the paper.

3 SEARCHING THE BOUNDARY OF THE FEASIBLE REGION

It is a common situation for many constrained optimization problems that some constraints are active at the target global optimum. Thus the optimum lies on the boundary of the feasible space. On the other hand, it is commonly acknowledged that restricting the size of the search space in evolutionary algorithms (as in most other search algorithms) is generally beneficial. Hence, it seems natural in the context of constrained optimization to restrict the search of the solution to some part of the boundary of the feasible part of the space, i.e., the part of R^n where some of the inequality constraints $g_i, i \in [1, q]$ of equation (1) actually are equalities.

We assume in the rest of the paper that we are searching a surface in R^n, be it the boundary between feasible and infeasible parts of the search space (in the case of inequality constraints g_i, $1 \leq i \leq q$), or the feasible part of the search space itself (as it is the case for equality constraints g_i, $q+1 \leq i \leq m$, which define a surface). In this study we suppose further that the surface is given as the n-dimensional unit-sphere, i.e., given as

$$\sum_{i=1}^{n} {x_i}^2 = 1.$$

The main interest of the sphere as a surface in R^n come from both its simple analytical expression and its nice symmetrical properties. Hence different methods to design evolution operators on the sphere can be used (see next section). However, most of the operators presented here can be generalized easily to operators acting on any regular (i.e., the gradient vector, orthogonal to the surface, is defined almost everywhere) Riemanian surface S of dimension $n-1$ in the space R^n, and even on any regular Riemanian surface of dimension $k < n$ —which is the case where more than one constraint g_i, $q+1 \leq i \leq m$ has to be taken into account. Additional motivation for investigating a surface defined by a sphere is taken from the observation that whenever the Hessian (derivatives of the gradient) is defined on the surface, this surface can be locally approximated at second order by an ellipsoid. All operators introduced in the next section can therefore be locally generalized.

An evolutionary system for searching the boundary between feasible and infeasible parts of the search space should initialize the population accordingly (all generated individuals should lie on the boundary) and the boundary operators should transform one or more boundary points into a new boundary point. The initialization process is conceptually quite simple: a sphere should be sampled as uniformly as possible according to the distance at hand. The exact initialization method depends on particular representation of potential solutions. For example, if a (randomly generated) solution is represented as n-dimensional vector of Euclidean coordinates $(a_1, a_2, \ldots, a_n)$, then the initialization procedure may use a *repair* operator, which divides the coordinates of the vector by its Euclidian norm:

$$p = (a_1/s, a_2/s, \ldots, a_n/s), \tag{2}$$

where $s = \sqrt{\sum_{i=1}^{n} a_i^2}$.

For a parametric representation of a surface, it is also easy to design a simple initialization algorithm. Suppose the surface S (of dimension $n-1$) is defined by $x_i = s_i(t_1, \ldots t_{n-1})$, $t_i \in [a_i, b_i]$, for $i = 1, \ldots, n$, where the functions s_i are regular functions from R^{n-1} into R. Now, a random choice of a point on S amounts to the choice of the $n-1$ values of the parameters $t_1, \ldots t_{n-1}$, uniformly on $\Pi[a_i, b_i]$.

For instance, the well known *spherical coordinates* give such a parametric representation of the sphere:

$$\left\{ \begin{array}{lll} x_1 & = & \cos(t_1) \\ x_2 & = & \sin(t_1)\cos(t_2) \\ \ldots & & \\ x_{n-1} & = & \sin(t_1)\ldots\sin(t_{n-2})\cos(t_{n-1}) \\ x_n & = & \sin(t_1)\ldots\sin(t_{n-2})\sin(t_{n-1}) \end{array} \right. \tag{3}$$

with $t_i \in [0, \pi]$ for $1 \leq i \leq n-2$ and $t_{n-1} \in [-\pi, \pi]$ (or $t_i \in [0, \pi/2]$ if all Euclidean coordinates need to be positive.

Assuming that the initial population consists of points on the surface, we need to design *closed* boundary operators which transform point(s) of the surface into point(s) of the surface. It is desirable that these boundary operators respect some experimentally and empirically derived properties [Rad91]:

- recombination should be able to generate all points "between" the parents;

- mutation should be *ergodic*, having non-zero probability to reach any point within a finite number of application, and should respect the principle of *strong causality* [Rec73], i.e., small mutations must result in small changes in the fitness function.

4 BOUNDARY OPERATORS

In this section we discuss several classes of surface operators on a sphere.

4.1 USE OF REPAIR OPERATOR

Note that if potential solutions are represented as vectors of Euclidean coordinates, then one can apply any operators (intermediate recombination, Gaussian mutation, etc.) followed by a repair operator, which (as discussed in the previous section) normalizes coordinates of an offspring, moving it to the sphere. Note, that the repair operator may follow any crossover and any mutation—this approach demonstrates effective use of repair algorithms in evolutionary systems.

4.2 SPHERE OPERATORS

Another possibility is to design a specialized operators which, by themselves—without any repair procedure, create a new boundary point. The *sphere crossover* is an example of such approach: it produces one offspring (z_i) from two parents (x_i) and (y_i) by:

> $z_i = \sqrt{\alpha x_i^2 + (1-\alpha) y_i^2}$, $i \in [1, n]$, with α randomly chosen in $[0, 1]$.

Note, that this operator can be generalized easily into k-parent sphere crossover (parents: $x^1, x^2, \ldots, x^k$) by

$$x_i^{k+1} = \sqrt{\alpha_1 (x_i^1)^2 + \ldots + \alpha_k (x_i^k)^2},$$

for non-negative α_i such that $\alpha_1 + \ldots + \alpha_k = 1$.

Similarly, the following sphere-specific mutation transforms (x_i) by selecting two indices $i \neq j$ and a random number p in $\langle 0, 1 \rangle$, and setting:

> $x_i \rightarrow p \cdot x_i$ and $x_j \rightarrow q \cdot x_j$,
> where $q = \sqrt{(\frac{x_i}{x_j})^2 (1 - p^2) + 1}$.

4.3 CURVE OPERATORS

Additional method to design suitable operators is based on *curves* drawn on the surface. From curves joining two different points, one can derive a crossover operator by choosing as offspring one (two) point(s) on that curve. Minimal-length curves (termed *geodesical curves*) seem a priori a good choice: their existence is guaranteed, locally on any regular surface from standard Cauchy-Lipschitz theorem, and globally (i.e., joining any pair of points of the surface) if the surface is *geodesically complete*[1] (Hopf-Rinov theorem) [Mar90]. Moreover, in the linear case, the geodesical curve between two points is the line segment between them, and the curve-based operator is nothing but the standard linear recombination operator [Sch95]. But recent experiments suggest that other paths between both parents can be used successfully [SM96]: the minimal length requirement does not seem mandatory.

From a beam of curves starting from one point, one can derive a mutation operator by first choosing randomly one curve in the beam, then choosing a point on the chosen curve. A desirable property of the beam of curves related to the ergodicity of the resulting mutation operator is that a large enough neighborhood of the starting point is covered by such set of curves: the local geodesical curves defined from the parent point and one tangent direction are such sets, defined almost everywhere on regular surfaces [Mar90]. On the sphere, for instance, the geodesical curves from a pole are the meridians, which in that case cover not only a whole neighborhood of the parent point, but the whole surface. Furthermore, a tight control of the distance between parents and offspring allows for a simple implementation of adaptive mutation respecting the strong causality principle.

Unfortunately, in the general case, even with analytical definitions of the constraints, the derivation of the geodesical curves (or the exponential curves beam) is not tractable: it involves heavy symbolic computations, plus the numerical solution of many local second-order systems of differential equations. Moreover, unavoidable numerical errors would probably move the solutions off the desired curves.

The case of the unit sphere, however, is an exception. The computation of the geodesical curve is straightforward. The geodesical crossover uses the circle going through both parents. For the mutation, a random vector is chosen, orthogonal to the gradient at the parent point. The offspring is then chosen on the

[1] A surface is geodesically complete if no geodesical curve encounters a dead-end.

geodesical circle tangent to this vector; the distance from parent to offspring follows a Gaussian law (whose standard deviations can be eventually adjusted adaptively).

However, in some other situations, the general ideas presented above can be implemented without the need for exact geodesical curves. The sphere operators of section 3 constitute a particular example of a such curve.

4.4 PLANE OPERATORS

Another general method to design curves, and hence operators to evolve on a surface of dimension $n-1$ in R^n, is to use the intersection of that surface with 2-dimensional planes.

Consider two points A and B belonging to S, and a vector $\vec{v}$ which is not collinear to $\vec{AB}$ and which is not orthogonal to the gradient vector at point A. Hence the plane defined by $(A, \vec{AB}, \vec{v})$ intersects the surface around A, defining a curve on S. If this curve connects points A and B, an appropriate crossover operator can be designed. But this procedure can fail if the intersection is not connected (as for the "horse-shoe sausage" of R^3 with A and B at both ends).

Similarly, the mutation operator can be designed by choosing the gradient at point A instead of vector $\vec{AB}$ above, with a prescribed distance from parent point A. Examples of plane-based operators will now be given in the simple case where S is a sphere.

In the case of the sphere, the derivation of curves joining two points by intersecting the sphere with a plane results in straightforward calculation. Moreover, the geodesical curves are particular cases of such plane-based curves, corresponding to the case where the chosen plane goes through the center of the sphere.

4.5 PARAMETRIC OPERATORS

For a parametric representation of a surface, it is easy to design closed operators. Suppose the surface S (of dimension $k < n$) is defined by $x_i = s_i(t_1, \ldots t_k)$, $t_i \in [a_i, b_i]$, for $i = 1, \ldots, k$, where the functions s_i are regular functions from R^k into R. Now, a random choice of a point on S amounts to the choice of the k values of the parameters $t_1, \ldots t_k$, uniformly on $\Pi[a_i, b_i]$.

In the following, we denote this relation $(x_i) = S[(t_i)]$.

The crossover can be defined by:

$$S[(t_i)], S[(u_i)] \rightarrow S[(\alpha t_i + (1-\alpha) u_i)],$$

for some α randomly chosen in $[0, 1]$.

Similarly, the mutation can be given by

$$S[(t_i)] \rightarrow S[(t_i + N(0, \sigma_i))],$$

where $N(0, \sigma)$ denotes the normal distribution with zero mean and σ variance. The parameters σ_i can be either user-supplied (eventually dynamically) or adaptive, i.e., encoded in the individual, as in evolution strategies.

For the case ot the unit sphere, these formulas can be applied to the spherical coordinates given by equation (3). In that context, the choice of a specific parametric representation fully determines the operators. Note that there are $n!$ valid parametric representations for the sphere, corresponding to the permutations among coordinates. Figure 1 gives an example of three curves defining these operators on the sphere in the case $n = 3$ (denoted P_1, P_2, and P_3), together with the geodesical curve (denoted G) and the curve corresponding to the spherical crossover (denoted Sp).

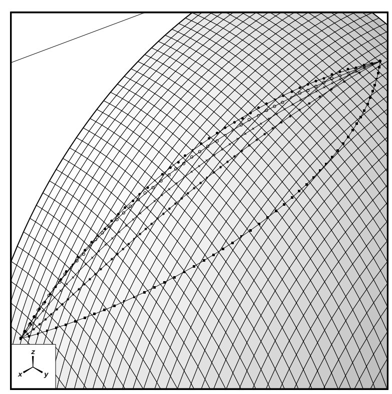

Figure 1: Crossover operators on the sphere. From top to bottom, P_1, P_2, G, Sp, P_3, where P_i correspond to 3 different spherical parametric representation, G to the geodesical curve and Sp to the specific sphere-operators

5 TEST CASES, EXPERIMENTS AND RESULTS

This section presents experimental comparative results involving some operators discussed in the previous section.

5.1 EXPERIMENTAL CONDITIONS

The following operators were considered:

- Standard intermediate crossover and Gaussian mutation (with adaptive standard deviations

[Sch95]) on Euclidian coordinates followed by the repair operator given by equation (2).

- The specialized spherical operators described in section 4.2.
- The operators based on geodesical curves (section 4.3). Note that the geodesical curve between two points is the repaired segment between there two points. Hence, the crossover is equivalent to the repair crossover with a slightly different probability distribution of offspring along the circle.
- The plane-based operators described in section 4.4. Note that, on the sphere, the geodesical operators are a special case of plane-based operators.
- Standard intermediate crossover and Gaussian mutation (again with adaptive standard deviations) on spherical parametrical representation, as given by equation (3).

In order to try to determine the respective influence of mutation and crossover, two different evolution schemes were considered, with varying probabilities of applying operators:

- a generational genetic algorithm (floating-point representation) with 2-tournament selection, population size of 60, crossover probability of 0.6, and mutation probability of 0.2 per individual (i.e. when an individual undergoes mutation, all its coordinates are modified following a Gaussian law).
- $(7, 60)$–ES, where 7 individual give birth to 60 offspring by mutation only, among which the 7 best are retained to become the parents of next generation.

5.2 TEST CASES

All considered test cases are defined on R^n with constraints $\sum_{i=1}^{n} x_i^2 = 1$ and $0 \leq x_i \leq 1$ for $1 \leq i \leq n$.

The first test case considered in this paper is to maximize [SM96]

$$P(\vec{x}) = (\sqrt{n})^n \cdot \prod_{i=1}^{n} x_i,$$

The function has a global solution at $(x_1, \ldots, x_n) = (\frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}})$ and the value of the function in this point is 1.

The second test case is to maximize [Bal95]:

$$B_1(\vec{x}) = \frac{100}{0.00001 + \sum_{i=1}^{n} |y_i|},$$

where $y_i = x_i + y_{i-1}$ for $i = 1, \ldots, n$ with $y_0 = 0$, $\sum_{i=1}^{n} x_i^2 = 1$. On the sphere with positive coordinates, the function has a global solution at $(x_1, \ldots, x_n) = (0, 0, \ldots, 0, 1)$ and the value of the function in this point is 99.999 (the original function in [Bal95] was defined for $-2.56 \leq x_i \leq 2.56$ and reached its maximum value of 10^7 at point (0,0,...0)). This function was chosen for its difficulty, at least in the unconstrained case.

The third test case is to maximize [Bal95] B_2 (which is expressed by the same formula as B_1, except $y_i = x_i + \sin(y_{i-1})$ for $i = 1, \ldots, n$ with $y_0 = 0$. As for function B_1, the function has a global solution at $(x_1, \ldots, x_n) = (0, 0, \ldots, 0, 1)$ and the value of the function in this point is 99.999.

5.3 RESULTS

Product function P

The first test function P is relatively simple: all its variables can be set independently, but for the constraint of being on the sphere.

The first three comparison experiments were done for various number of variables of the function, namely $n = 50$, 100 and 200. The performance of the GA described in section 5.1 using various boundary operators for $n = 100$ is displayed in Figure 2; the plots for $n = 50$ and $n = 200$ were almost identical. It is clear that the spherical operators are the best, with plane and geodesical operators only slightly worse; on the other hand, repair and parametric operators gave very poor performance.

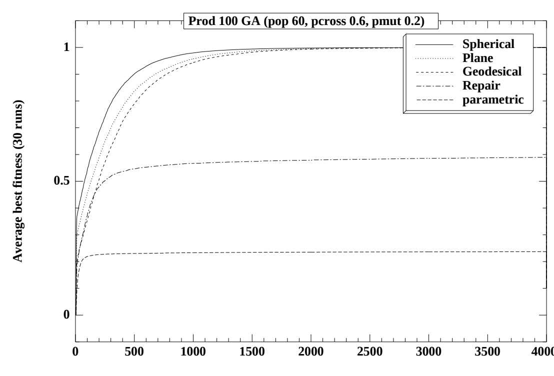

Figure 2: Comparison of various boundary operators; GA on P with $n = 100$

To differentiate the roles of mutations vs. crossovers (for each category of these boundary operators), we run the ES described in section 5.1 with mutation only. The results were quite poor (see Figure 3 for the test case of $n = 100$).

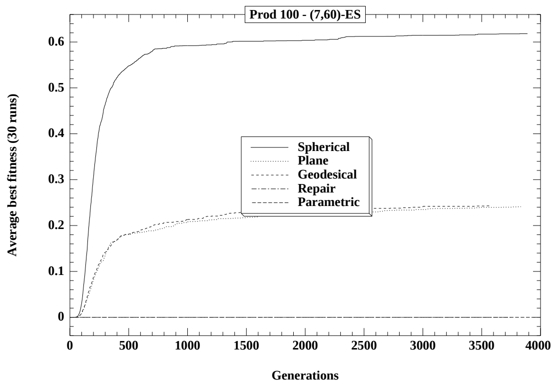

Figure 3: Comparison of various boundary operators; ES on P with $n = 100$

For a fair comparison, additional runs were made: the GA was used with probability of mutation 1 and probability of crossover 0 and 0.6; in both cases the performance of the system was much worse than for GA reported in Figure 2. Also, the ES (Figure 3) *with* the boundary crossover operator (probability of crossover being 0.6), showed improved performance (Figure 4). The above experiments demonstrate clearly the importance of boundary crossovers!

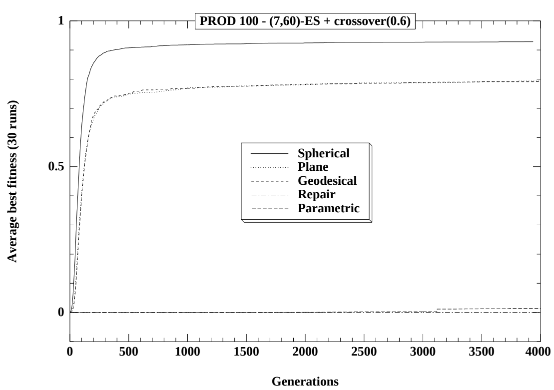

Figure 4: Comparison of various boundary operators; ES on P with $n = 100$ with boundary crossovers

Note, also, that the relationship between boundary operators was the same in all cases: spherical operators were the best, followed closely by plane and geodesical operators, with repair and parametric operators performing poorly.

Baluja's functions B_1 and B_2

The same series of experiments were run for the B_1 and B_2 functions. Both functions B_1 and B_2 gave very similar results, so only results on B_1 are presented here.

For both functions, the general picture is rather different. Still, the spherical mutation seems to be *the* operator designed to optimize functions B_1 and B_2: its effect is to move some "weight" from one coordinate to another, "knowing" that the sum of all squares remains constant. Hence, half of spherical mutations result in an improvement of the fitness (when the coordinate of a higer index is increased). So it is not surprising that the spherical mutation alone gives the best results (whether using GA or ES selection)—see Figures 5 and 6.

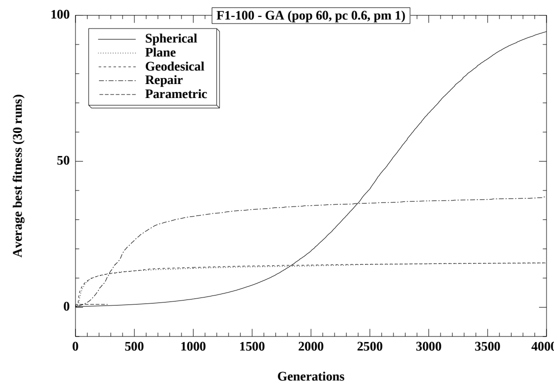

Figure 5: Comparison of various boundary operators; GA on B_1 with $n = 100$

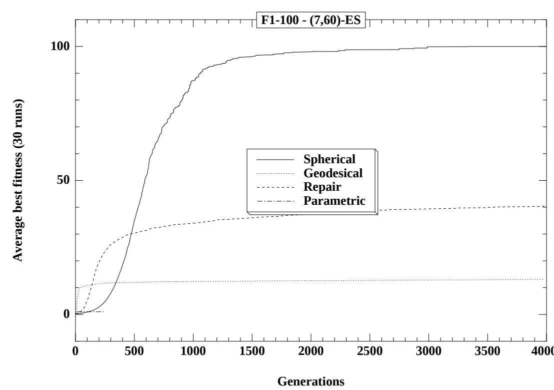

Figure 6: Comparison of various boundary operators; ES on B_1 with $n = 100$

However, it is interesting to note that
– adding crossover degrades the performance of GA and ES,
– even for other boundary operators, boundary mutation seems always better than the corresponding

crossover,
– standard operators plus repair operator gave good results—in comparison with other operators—though far behind those of spherical mutation.

Rotated Baluja functions

To eliminate the bias favoring spherical mutation in test cases B_1 and B_2, we slightly modified B_1 functions using rotations centered at the origin. Thus the coordinates are somehow intermixed.

The first interesting result regards the parametric representation given by equation (3): in all previous experiments, the parametric representation gave the worst results. However, a close examination of these results on function B_1 showed that the population rapidly converged to the local optimum $E_1 = (1, 0, 0, ..., 0)$. The appealing experiments were to put the global optimum first at E_1, then at different E_i. Figure 7 shows the results of parametric representation in such cases for $i = 1, \ldots, 6$, and shows that indeed parametric representation performances highly depend on the order in which the parameters are considered in equation 3.

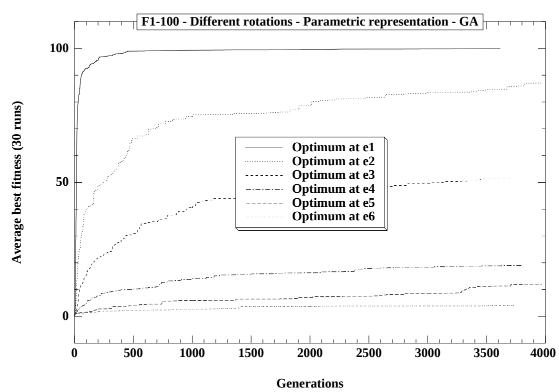

Figure 7: Parametric representation with different rotations; B_1 with $n = 100$

Two other rotations were also experimented with: the former one brought the optimum to $(1/\sqrt{(n)}, \ldots, 1/\sqrt{(n)})$ while the optimum of the latter one was randomly chosen with positive coordinates (see Tables 1 and 2, respectively). Here, the results are "back to normal": sphere operators being the best; boundary crossover being of importance.

Tables 1 and 2 provide with a rich experimental material. Note that in the case with the optimum on the diagonal (Table 1), GA with parameters 0.6 and 0.2 (for probabilities of crossover and mutation) outper-

	Sph.	Pl.	Geo.	Rep.	Par
GA	61.7	54.24	48.66	53.38	38.99
(0.6,0.2)	(3.66)	(4.99)	(4.12)	(3.93)	(3.47)
GA-M$^+$	53.29	44.47	37.70	15.60	15.91
(0.6,1)	(2.06)	(2.82)	(2.83)	(1.03)	(4.44)
GA-¬X	27.57	26.71	26.22	15.69	14.15
(0,1)	(1.56)	(1.87)	(2.36)	(1.39)	(6.12)
ES	29.59	27.25	27.18	16.27	12.62
(0,1)	(1.36)	(1.74)	(1.38)	(1.25)	(4.76)
ES-X	38.88	41.27	37.32	15.77	13.37
(0.6,1)	(2.96)	(2.96)	(2.52)	(0.88)	(5.82)

Table 1: Average off-line results (std. deviation) of different operators for different evolution schemes (crossover rate, mutation rate) on function B_1 rotated such that the optimum is at on the diagonal

	Sph.	Pl.	Geo.	Rep.	Par
GA	30.48	27.23	25.76	21.75	13.99
(0.6,0.2)	(1.83)	(4.03)	(40.50)	(1.98)	(2.27)
GA-M$^+$	31.64	27.00	26.21	14.90	15.67
(0.6,1)	(1.3)	(1.81)	(1.97)	(1.85)	(6.40)
GA-¬X	25.45	22.31	22.43	15.30	15.33
(0,1)	(1.35)	(1.70)	(1.83)	(3.33)	(9.12)
ES	27.41	24.61	25.17	14.70	12.20
(0,1)	(1.33)	(1.24)	(1.17)	(0.59)	(4.93)
ES-X	29.71	32.35	30.86	14.43	13.33
(0.6,1)	(2.15)	(2.54)	(2.32)	(0.57)	(4.42)

Table 2: Average off-line results (std. deviation) of different operators for different evolution schemes (crossover rate, mutation rate) on function B_1 randomly rotated

forms all other schemes. Also, in this test case repair operators are better than plane and geodesical ones. In the case of random rotation (Table 2), if ES with crossover is used, plane operators perform better than sphere operators!

6 CONCLUSIONS

The results of experiments indicate that different boundary operators for different test functions, may give different performance. It seems that for the test function P the boundary crossover is of great importance (e.g., ES with boundary crossover is far better than ES without; GA with probability of crossover 0.6 and mutation 0.2 is far ahead from all other settings. Moreover, the increase in mutation rates decreases the performance of GA).

On the other hand, the test functions B_1 and B_2 yield just the opposite results: the optimum is reached only when the probability of mutation is set to 1; addition of boundary crossover rather harms the algorithm (whether GA or ES) than helps; ES was much better than GA. However, the performance of spherical mutation can be easily explained (see section 5.3); the results of experiments made on rotated B_1 and B_2 confirm the performance hierarchy of boundary operators similar to this of the test function P. The parametric representation is quite sensitive on the order in which the parameters are considered in equation 3.

It is interesting to note the weak performance of the standard operators undergoing repair. It can be easily understood why the repaired mutation can fail: even if the self-adaptive mechanism works perfectly, the standard mutation will select an offspring out of the sphere. The repair operator moves the offspring closer to the parent (it can even be repaired to the parent position). So the self-adaptive mutation mechanism is biased twice: first, because the distance between offspring and parent does not have the same probability distribution than for unconstrained problems; second, because of the first, the feed-back on self-adaptiveness is biased too, resulting in wrong selection criterion.

All other adaptive mechanism for the standard deviations we tried on the repair algorithm (one standard deviation per individual, or one standard deviation for the population updated via the one-fifth rule) proved much worse than the one used here (self-adaptation of one standard deviation per coordinate). On the other hand, the mutations for both the plane operators and the geodesical operators only used one self-adaptive standard deviation for the distance parent-offspring, the direction of mutation being randomly chosen.

This suggest to design yet another mutation operator, that would benefit from both features: use one standard deviation parameter for each variable, thus determining the *direction* of mutation, and have an additional—self-adaptive—parameter to control the distance from parent to offspring.

Further study will investigate further complex connections between characteristic of the objective function, evolutionary scheme used, adaptiveness, and boundary operators, as well as study other (than sphere) surfaces.

Acknowledgments

The second author acknowledges the support of the National Science Foundation under Grant IRI-9322400.

References

[Bal95] S. Baluja. An empirical comparison of seven iterative and evolutionary function optimization heuristics. Technical Report CMU-CS-95-193, School of Computer Science, Carnegie Mellon University, 1995.

[Mar90] C. Margerin. Une introduction à la géométrie. Course at ENSTA – Palaiseau – France, 1990.

[MA94] Z. Michalewicz, T.D. Logan, and S. Swaminathan. Evolutionary Operators for Continuous Convex Parameter Spaces. In *Proceedings of the 3rd Annual Conference on Evolutionary Programming*, pages 84–97, 1994.

[MS96] Z. Michalewicz and M. Schoenauer. Evolutionary Algorithms for Constrained Parameter Optimization Problems. *Evolutionary Computation*, Vol.4, No.1, 1996, pp.1–32.

[Rad91] N. J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–20, 1991.

[Rec73] I. Rechenberg. *Evolutionstrategie: Optimierung Technisher Systeme nach Prinzipien des Biologischen Evolution.* Stuttgart: Fromman-Holzboog Verlag, 1973.

[SM96] M. Schoenauer and Z. Michalewicz. Evolutionary Computation at the Edge of Feasibility. In H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel (Editors), *Proceedings of the 4th Parallel Problem Solving from Nature*, Berlin, September 22–27, 1996, Springer-Verlag, Lecture Notes in Computer Science, Vol.1141, pp.245–254.

[Sch95] H.-P. Schwefel. *Evolution and Optimum Seeking.* John Wiley, Chichester, UK, 1995.