

# BBOB-Benchmarking the Rosenbrock's Local Search Algorithm

Petr Pošík

Czech Technical University in Prague, Faculty of Electrical Engineering, Dept. of Cybernetics  
Technická 2, 166 27 Prague 6  
posik@labe.felk.cvut.cz

## ABSTRACT

The restarted Rosenbrock's optimization algorithm is tested on the BBOB 2009 testbed. The algorithm turned out to be very efficient for functions with simple structure (independently of dimensionality), but is not reliable for multimodal or ill-conditioned functions.

## Categories and Subject Descriptors

G.1.6 [Numerical Analysis]: Optimization—*Global Optimization, Unconstrained Optimization*; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems

## General Terms

Algorithms, Experimentation, Performance, Reliability

## Keywords

Benchmarking, Black-box optimization, Local search, Rosenbrock's algorithm, Evolutionary computation

## 1. INTRODUCTION

The Rosenbrock's algorithm [6] is a classical local search technique—it maintains the best-so-far solution and searches in its neighborhood for improvements. What distinguishes this algorithm from other local search techniques is the fact that it also maintains a model of the current local neighborhood—it adapts the model orientation and size. This feature can be observed in many recent successful optimization techniques, e.g. in CMA-ES [4].

## 2. ALGORITHM DESCRIPTION

The Rosenbrock's local search technique is depicted as Alg. 1. The model of the local neighborhood consists of  $D$  vectors  $\mathbf{e}_1, \dots, \mathbf{e}_D$  forming the orthonormal basis, and of  $D$  multipliers (or step lengths, if you want)  $d_1, \dots, d_D$ , where

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GECCO'09, July 8–12, 2009, Montréal Québec, Canada.  
Copyright 2009 ACM 978-1-60558-505-5/09/07 ...\$5.00.

---

### Algorithm 1: Rosenbrock's Algorithm

---

```
Input:  $\alpha > 1, \beta \in (0, 1)$   
1 begin  
2    $\mathbf{x} \leftarrow \text{Initialize}(); \mathbf{x}_o \leftarrow \mathbf{x}$   
3    $\{\mathbf{e}_1, \dots, \mathbf{e}_D\} \leftarrow \text{InitOrthoBasis}()$   
4    $\{d_1, \dots, d_D\} \leftarrow \text{InitMultipliers}()$   
5   while not TerminationCondition() do  
6     for  $i=1..D$  do  
7        $\mathbf{y} \leftarrow \mathbf{x} + d_i \mathbf{e}_i$   
8       if BetterThan( $y, x$ ) then  
9          $\mathbf{x} \leftarrow \mathbf{y}$   
10         $d_i \leftarrow \alpha \cdot d_i$   
11       else  
12         $d_i \leftarrow -\beta \cdot d_i$   
13     if AtLeastOneSuccessInAllDirs() and  
        AtLeastOneFailInAllDirs() then  
14        $\{\mathbf{e}_1, \dots, \mathbf{e}_D\} \leftarrow \text{UpdOrthoBasis}(\mathbf{x} - \mathbf{x}_o)$   
15        $\{d_1, \dots, d_D\} \leftarrow \text{InitMultipliers}()$   
16        $\mathbf{x}_o \leftarrow \mathbf{x}$   
17 end
```

---

$D$  is the dimensionality of the search space. In each iteration, the algorithm performs a kind of pattern line search along the directions given by the orthonormal basis. If in one direction  $\mathbf{e}_i$  an improvement is found, next time (after trying all other directions) a point  $\alpha$  times further in that direction is sampled; if no improvement is found in the  $\mathbf{e}_i$  direction, next time a closer point on the other side is sampled (governed by the  $\beta$  parameter). Usually, the values of parameters are  $\alpha = 2$  and  $\beta = \frac{1}{2}$ .

As soon as at least one successful and one unsuccessful move in each direction was carried out, the algorithm updates its orthonormal basis to reflect the cumulative effect of all successful steps in all directions. It also resets the multipliers to their original values. The update of the orthonormal basis is done by the Palmer's orthogonalization method [5] so that the first basis vector is always parallel to the last vector  $\mathbf{x} - \mathbf{x}_o$ .

The demonstration of the Rosenbrock's algorithm behaviour on 2D sphere and 2D Rosenbrock's function can be seen in Fig. 1.

To improve performance on multimodal functions, a restarting strategy (see Fig. 2) was used. Each restart begins with an initial point uniformly chosen from the interval  $(-5, 5)^D$ .

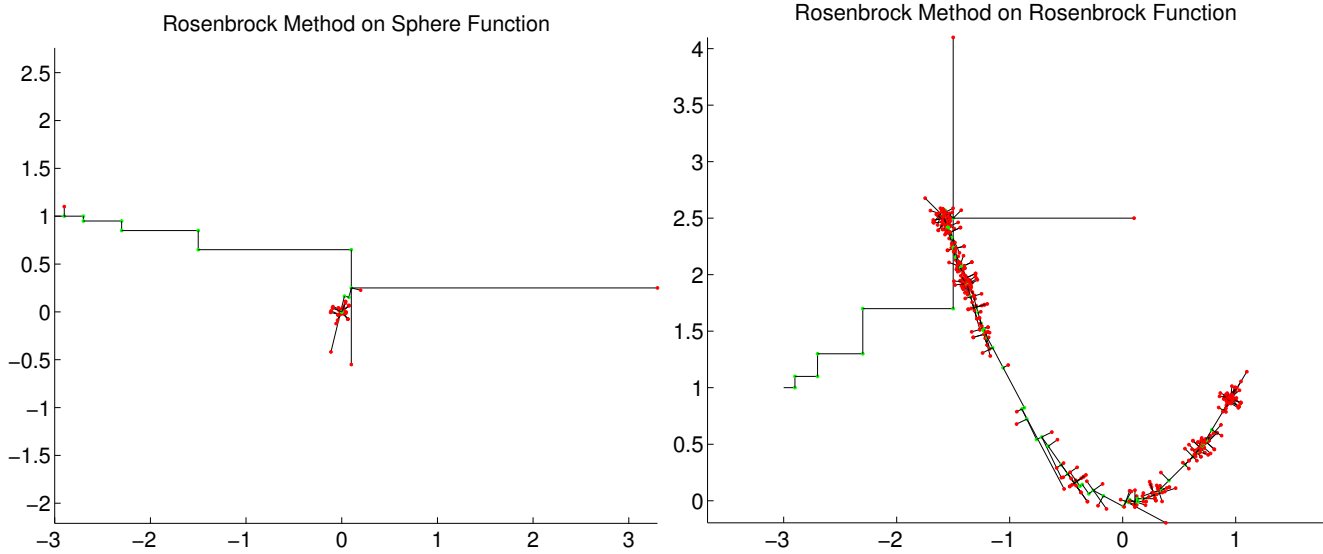


Figure 1: The behaviour of the Rosenbrock's optimization algorithm on the sphere (left) and on the Rosenbrock's (right) function

---

```

function [x, ilaunch] = bbobRestRosenbrockLS(FUN, ...
    DIM, ftarget, outermaxfunevals, innermaxfunevals, varargin)

options = struct('MaxFunEvals', min(1e8*DIM, outermaxfunevals), ...
    'MaxIter', inf, ...
    'Tolfun', 1e-11, ...
    'TolX', 1e-9, ...
    'StopFitness', ftarget, ...
    'Display', 'off' ...
);

for ilaunch = 1:100; % relaunch optimizer up to 100 times
    % set initial conditions
    ropts = options;
    ropts.MaxFunEvals = min( innermaxfunevals, outermaxfunevals - feval(FUN, 'evaluations'));
    ropts.dInit = 0.1;

    xstart = -5 + 10 * rand(1,DIM); % random start solution

    % try the Rosenbrock's algorithm
    x = minNDRosenbrockMod(FUN, xstart, ropts);
    if feval(FUN, 'fbest') < ftarget || feval(FUN, 'evaluations') >= options.MaxFunEvals,
        break;
    end
end
end

```

---

Figure 2: Restarting procedure for the Rosenbrock's Algorithm

## 2.1 Implementation Modifications

The original Rosenbrock’s algorithm resets the multipliers  $d_i$  at each stage of the algorithm, i.e. each time the orthonormal basis is updated. In the particular implementation used in this article, the multipliers are not reset. It was observed that this modification improves the results on many benchmark problems—it spares some function evaluation needed to adapt the multipliers at each stage. Instead, it converges faster, allowing for more algorithm restarts.

## 2.2 Parameter Settings

The algorithm has 2 parameters,  $\alpha$  and  $\beta$ . As already stated, the default values of  $\alpha = 2$  and  $\beta = \frac{1}{2}$  were used.

## 2.3 Box Constraints? No

The algorithm was run in unconstrained setting. In the BBOB comparison, some global search algorithms are present and these usually need the box constraints to be able to work. In this sense, the comparison is not quite fair since algorithms run in the unconstrained setting may spend some part of the available evaluation budget evaluating solutions lying far away from the area of the global optimum.

## 2.4 The Crafting Effort

The algorithm has no additional parameters, so that no further tuning is necessary; the crafting effort CrR = 0.

## 2.5 Invariance Properties

With the exception of initialization, the algorithm is *invariant with respect to translation and rotation* which is demonstrated in the next section on the unrotated and rotated Rastrigin functions (3 and 15), and the Rosenbrock function (8 and 9). Looking at the separable and non-separable version of the ellipsoid function (2 and 10), we can see rather unexpected phenomenon: a different pattern which would suggest that the algorithm is not rotationally invariant. For the time being, we do not have any sound explanation for this effect, but we believe it is caused by the initialization which always initializes the algorithm with the best possible model in case of separable Ellipsoid function, while it initializes the algorithm with a bad model in case of non-separable Ellipsoid function.

The algorithm is also *invariant with respect to order-preserving transformations of the fitness function* since it uses only comparisons between two individuals.

## 3. EXPERIMENTAL PROCEDURE

The standard experimental procedure of BBOB was adopted: the algorithm was run on 24 test functions, 5 instances each, 3 runs on each instance. Each run was finished

- after finding a solution with fitness difference  $\Delta f \leq 10^{-8}$ , or
- after performing more than  $10^4 \times D$  function evaluations.

Each individual launch of the basic Rosenbrock’s algorithm was interrupted (and the algorithm was restarted)

- after finding a solution with fitness difference  $\Delta f \leq 10^{-8}$ , or
- after performing more than the allowed number of function evaluations, or

- after the model converged too much, i.e. when  $\max_i |d_i| < 10^{-9}$ .

## 4. RESULTS

Results from experiments according to [2] on the benchmark functions given in [1, 3] are presented in Figures 3 and 4 and in Table 1. The method solves at least once (out of 15 runs) 20 (2D), 16 (3D), 13 (5D), 8 (10D), and 5 (20D) out of 24 functions.

## 5. CPU TIMING EXPERIMENT

The multistart algorithm was run with the maximal number of evaluations set to  $10^5$ , the basic algorithm was restarted for at least 30 seconds. The algorithm takes on average from  $3.1 \cdot 10^{-4}$  to  $3.7 \cdot 10^{-4}$  seconds per function evaluation for all tested dimensionalities (2–40) of the search space. The experiment was conducted on Intel Core 2 CPU, T5600, 1.83 GHz, 1 GB RAM with Windows XP SP3 in MATLAB R2007b.

## 6. CONCLUSIONS

The restarted Rosenbrock’s optimization algorithm is an optimization technique suitable for functions with simple structure and low number of local optima. For the most simple functions (Sphere, Linear slope), the algorithm finds the solution very quickly and exhibits only linear scaling. The more complex functions (especially highly multimodal or ill-conditioned) are very hard for this algorithm and it cannot be considered a reliable solver for them.

## Acknowledgements

The project was supported by the Ministry of Education, Youth and Sport of the Czech Republic with the grant No. MSM6840770012 entitled “Transdisciplinary Research in Biomedical Engineering II”.

## 7. REFERENCES

- [1] S. Finck, N. Hansen, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Presentation of the noiseless functions. Technical Report 2009/20, Research Center PPE, 2009.
- [2] N. Hansen, A. Auger, S. Finck, and R. Ros. Real-parameter black-box optimization benchmarking 2009: Experimental setup. Technical Report RR-6828, INRIA, 2009.
- [3] N. Hansen, S. Finck, R. Ros, and A. Auger. Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Technical Report RR-6829, INRIA, 2009.
- [4] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.
- [5] J. R. Palmer. An improved procedure for orthogonalising the search vectors in Rosenbrock’s and Swann’s direct search optimisation methods. *The Computer Journal*, 12(1):69–71, February 1969.
- [6] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, March 1960.

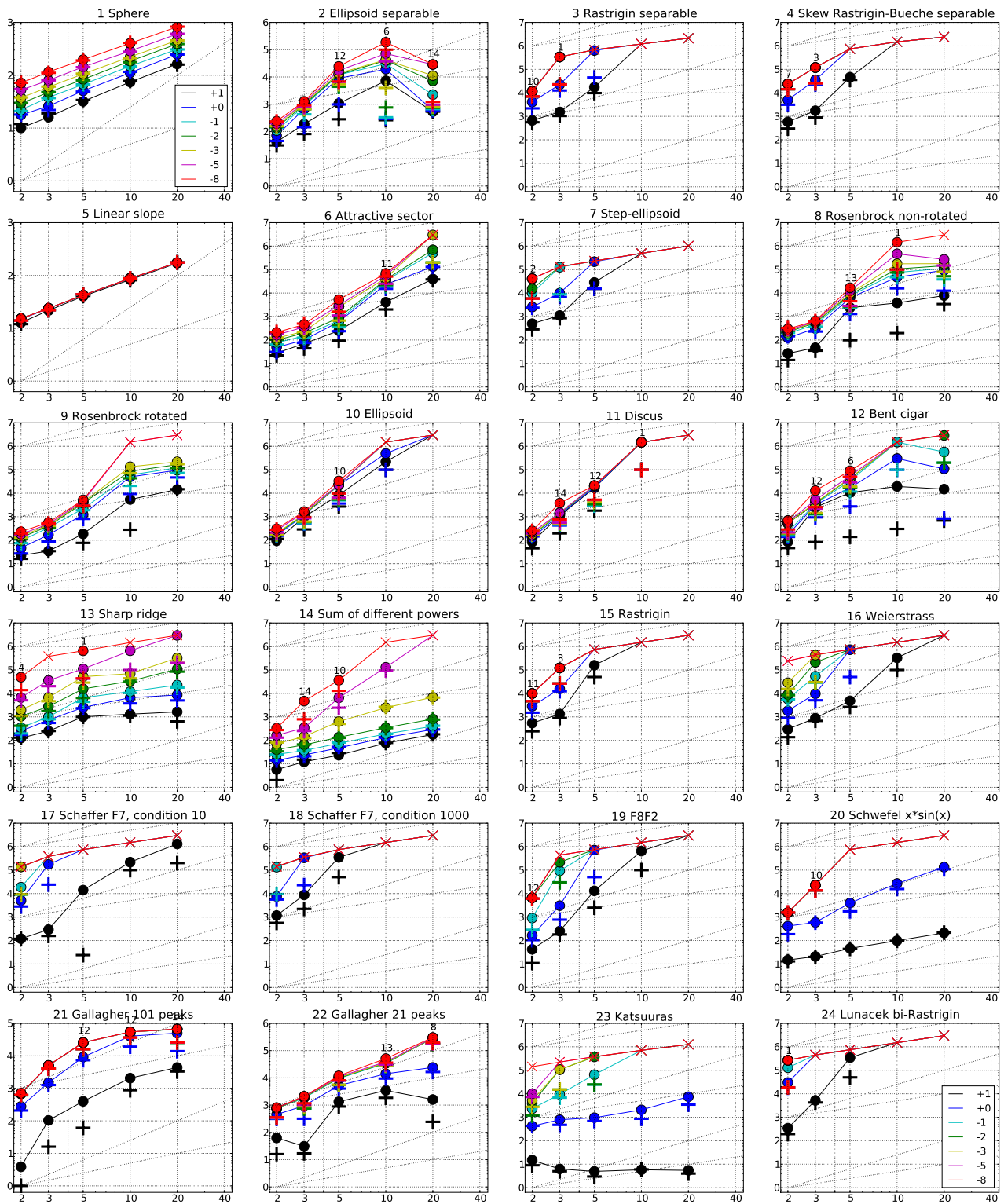


Figure 3: Expected Running Time (ERT, ●) to reach  $f_{\text{opt}} + \Delta f$  and median number of function evaluations of successful trials (+), shown for  $\Delta f = 10, 1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-5}, 10^{-8}$  (the exponent is given in the legend of  $f_1$  and  $f_{24}$ ) versus dimension in log-log presentation. The  $\text{ERT}(\Delta f)$  equals to  $\#FES(\Delta f)$  divided by the number of successful trials, where a trial is successful if  $f_{\text{opt}} + \Delta f$  was surpassed during the trial. The  $\#FES(\Delta f)$  are the total number of function evaluations while  $f_{\text{opt}} + \Delta f$  was not surpassed during the trial from all respective trials (successful and unsuccessful), and  $f_{\text{opt}}$  denotes the optimal function value. Crosses (×) indicate the total number of function evaluations  $\#FES(-\infty)$ . Numbers above ERT-symbols indicate the number of successful trials. Annotated numbers on the ordinate are decimal logarithms. Additional grid lines show linear and quadratic scaling.



