On a simple Markov homogeneous random search algorithm of an extremum

Alexey Tikhomirov[*]

July 27, 2022

Abstract

A simple Markov homogeneous random search algorithm of an extremumis is presented. This algorithm allows solving a fairly wide class of problems of finding the global extremum of an objective function with a high accuracy.

1 Introduction

Let the *objective function* $f\colon \mathbb{R}^d \mapsto \mathbb{R}$ take a minimum value at a single point x_*. Let us consider the problem of finding the global minimum point x_* with a given accuracy $\varepsilon > 0$. One way to solve this problem is to use random search algorithms for the extremum of a function (see [1]–[14]). Such methods have long been successfully used in solving complex optimization problems. Theoretical studies of the convergence rate of some Markov search algorithms are given in [3], [10]–[13]. This work is a continuation of [10] and is devoted to one simple but effective algorithm for a homogeneous Markov monotone search for an extremum.

We will consider the space $X = \mathbb{R}^d$ as an optimization space with metric

$$\rho(x, y) = \max_{1\leqslant n\leqslant d} |x_n - y_n|,$$

[*]Yaroslav-the-Wise Novgorod State University, Veliky Novgorod, Russian Federation, tikhomirov.as@mail.ru

where $x = (x_1, \dots, x_d)$ and $y = (y_1, \dots, y_d)$. A closed ball of radius r centered at x is denoted by $B_r(x) = \{y \in \mathbb{R}^d : \rho_\infty(x, y) \leqslant r\}$.

The metric ρ_∞ was chosen for reasons of simplicity of modeling the random search under consideration. The simulation of the considered random search is based on modeling uniform distributions in the balls. The ball in the metric ρ_∞ is a cube. Simulating a uniform distribution in a multidimensional cube is simple, while it is rather difficult to model a uniform distribution in a "regular" ball defined by the Euclidean metric.

To find the minimum point, we use a *homogeneous Markov monotone random search* described in [14].

2 Choosing search parameters

It is important to note that the choice of search parameters can have a major impact on the effectiveness of the random search method [3], [5], [7]. Moreover, many search algorithms contain a large number of heuristic parameters, and it is very difficult for the user of such an algorithm to find "good" parameter values that are suitable for the optimized function. Here is a quote from [7] related to the "very fast annealing method" proposed by L. Ingber: "Among the drawbacks of this method is that it sometimes takes several months to fine-tune it to solve a specific problem because a large number of parameters". Moreover, with the right selection of parameters, the "very fast annealing method" can show very good results [6], [7].

The presented search has only three parameters. Positive numbers ν and Γ determine the range of variation of the radii of the balls uniform distributions in which formed a transition function of the random search (see [14]). The third parameter is N — the number of search steps.

The value of ν can be chosen close to the required accuracy of the solution to the problem when approximated by argument. The value of Γ can be chosen close either to the assumed accuracy of the initial approximation (the distance from the initial search point to the minimum point) or to the diameter of the region under study in the optimization space. When choosing Γ, one can use the upper bound.

It is desirable to take the number of search steps N large enough. When solving a single task, you can, for example, complete a billion search steps, even if the task is simple enough and can be solved much faster. Modern personal computers may well perform such volumes of computations, at least for not too complex objective functions.

In addition to the three search parameters, it is necessary to select the starting point of the search. It is clear that the starting point is better located closer to the point of global extremum.

The proposed search algorithm is largely free from insurmountable difficulties in choosing parameters. In particular, in the numerical examples of section 4, the minimal choosing of parameters was performed, consisting of literally several attempts to run the program with different parameter values.

It should be noted that the simulation algorithm for the considered random search is very simple (see [14]).

3 Estimation of the complexity of the investigated search

Theoretical results on the complexity of the studied search can be obtained using the results of [10]. We use a random search to find the minimum point x_* with the given accuracy ε (approximation "by argument"). When approximating by argument, we will be interested in getting the search into the ball $B_\varepsilon(x_*)$.

We consider an analog of the considered random search from which the condition for stopping the search is removed for a theoretical study of the complexity of random search. Such an algorithm generates an infinite sequence of search points $\{\xi_k\}_{k=0}^{\infty}$. Through

$$\tau_\varepsilon = \min\{k \geqslant 0 : \xi_k \in B_\varepsilon(x_*)\}$$

we can denote the moment the search first hits the ε-neighborhood of the global minimum point. The distribution of the random variable τ_ε gives us quite complete information about the quality of the random search. Indeed, when performing the τ_ε steps of the search, the values of the objective function f are calculated $\tau_\varepsilon + 1$ times.

We consider one characteristic of the rate of convergence of a random search. The complexity of a random search is determined through $\mathsf{E}\,\tau_\varepsilon$ (mathematical expectation of τ_ε) and it makes sense the average number of search steps before it reaches the set $B_\varepsilon(x_*)$.

Using the results of [10], one can obtain a theoretical estimate of the complexity of the search under study. Let the objective function f be nondegenerate (see [10]). We choose the value of ν close to the required accuracy of the solution of the problem when approximating by argument (i.e., to the value ε). As a value Γ, we will take the upper estimate of the distance from the starting point of the search to the minimum point. Here, for a more accurate statement of the presented results, it would be necessary to introduce special concepts of [10]. Using the results of [10], we can show that $\mathsf{E}\,\tau_\varepsilon = O\left(\ln^2 \varepsilon\right)$.

For comparison, we can note that when using the simplest random search (the so-called "blind search" [3], [5]) which uses a uniform distribution in a pre-fixed area of the optimization space (and when optimizing non-degenerate objective functions) $\mathsf{E}\,\tau_\varepsilon = O\left(1/\varepsilon^d\right)$.

Thus, from a theoretical point of view, the studied search is fast (in the sense that its laboriousness has a good order of dependence on ε).

4 Examples of using the investigated random search

The program for numerical experiments is written in C# language. You can download the program at `www.novsu.ru/doc/study/tas1` from the "Random_search" folder. The program is available both in the form of an executable file, and in the form of a project containing the source code of the program and allowing the user to edit the program at their discretion.

The double numeric type, which provides an accuracy of 15–16 characters is used for calculations in he the program. Let us note that this numerical format limits the possible accuracy of the solution to the problem. Arguing somewhat simplistically, we obtain the following conclusions. If the objective function behaves approximately like a quadratic function in the vicinity of the global minimum, then with approximation accuracy with an argument of the order of 10^{-8} we obtain the approximation accuracy with respect to the value of a function of the order of 10^{-16}. If the minimum value of the objective function belongs to the interval $(1, 10)$, then the double numeric type, which provides an accuracy of 15–16 digits, will not allow calculating the value of the function with an accuracy above 10^{-16}. Thus, the possible accuracy of solving the problem will be of the order of 10^{-7} for approximation by argument, and of order 10^{-14} for approximation of the value of the function. Such accuracy, as a rule, is sufficient from a practical point of view. And such accuracy of solving the problem can be obtained by using the random search program in question when solving not too complicated optimization problems. Of course, if the minimum value of the objective function is equal to zero, and the minimum point is also at zero, then the problem can be solved with much higher accuracy.

Here are some examples of using the presented program to solve optimization problems. A personal computer with a processor Intel Core i5-4460S was used for calculations.

4.1 Example 1

Let us use the example from [5]. The optimization space is $X = \mathbb{R}^2$, $x = (x_1, x_2)$,

$$f(x) = f(x_1, x_2) = x_1^4 + x_1^2 + x_1 x_2 + x_2^2.$$

The function f takes a minimum value at a single point $x_* = (0, 0)$ and $f(x_*) = 0$. The starting point of the search is $x = (1, 1)$ and $f(x) = 4$. The number of search steps N is 10^4 here.

Algorithm B of the book [5] obtains the minimum value of the objective function 2.7×10^{-6}. Algorithm B corresponds to a homogeneous random search using the normal probability distribution as a transition function.

Algorithm C of the book [5] obtains the minimum value of the objective function 2.5×10^{-7}. Algorithm C also uses the normal probability distribution as a transition function, but is a more complex search option, in which, when constructing a new search point, the displacement made in the previous step of the algorithm is taken into account.

The considered random search with the parameters $\nu = 2 \times 10^{-24}$ and $\Gamma = 0.7$ receives the minimum value of the objective function 1.7×10^{-51}.

In this example, the considered random search turned out to be much more accurate than the algorithms B and C of the book [5], using the normal probability distribution.

The considered random search with the parameters $\nu = 10^{-165}$, $\Gamma = 0.7$, and $N = 10^6$ obtains the minimum value of the objective function equal to zero (i.e., less than the value 5×10^{-324}, defining a range of values of the type double of the C# programming language) and the minimum point $(-6.8 \times 10^{-163}, 1.7 \times 10^{-162})$. Let us note that in this case the maximum accuracy with which calculations in C# can be performed using the number format double (due to the fact that it is impossible to more accurately calculate the value of the objective function). The search execution time was

0.047 seconds.

4.2 Example 2

The optimization space here is $X = [-8, 8]^2$, $x = (x_1, x_2)$,

$$f(x) = f(x_1, x_2) = \frac{1}{2}\left((x_1^4 - 16x_1^2 + 5x_1) + (x_2^4 - 16x_2^2 + 5x_2)\right).$$

The function f has four local minima, one of which is global. The starting point of the search is $x = (4.0, 6.4)$ and $f(x) = 537.18$. The considered random search with parameters $\nu = 10^{-8}$, $\Gamma = 10$, and $N = 20000$ finds the minimum value of the objective function -78.3323314075428 and the minimum point $(-2.903534, -2.903534)$. Let us note that the extreme accuracy has been achieved with which you can perform calculations in C# using the double number format (due to the fact that it is impossible to calculate the value of the objective function more accurately).

4.3 Example 3

The space here is $X = [-4, 4]^{10}$, $x = (x_1, x_2, \dots, x_{10})$,

$$f(x) = f(x_1, x_2, \dots, x_{10}) = \sum_{n=1}^{5} \left(100(x_{2n} - x_{2n-1}^2)^2 + (1 - x_{2n-1})^2\right).$$

The f function is a well-known Rosenbrock test function used for local optimization methods. The function f takes the minimum value $f(x_*) = 0$ at the point $x_* = (1, 1, \dots, 1)$. The starting point of the search is $x = (-1.2, 1, -1.2, 1, \dots, 1)$ and $f(x) = 121$. The considered random search with parameters $\nu = 10^{-17}$, $\Gamma = 4$, and $N = 10^7$ finds the minimum value of the objective function 2.7×10^{-29}. The search time was 1.6 seconds.

4.4 Example 4

Consider an example with a very simple objective function, but in a very large dimension optimization space for random search methods. Here the space $X = \mathbb{R}^{1000}$, $x = (x_1, x_2, \dots, x_{1000})$,

$$f(x) = f(x_1, x_2, \dots, x_{1000}) = \sum_{n=1}^{1000} x_n^2.$$

The function f takes the minimum value $f(x_*) = 0$ at a single point. The starting point of the search is $x = (1, 1, \ldots, 1)$. The considered random search with parameters $\nu = 10^{-10}$, $\Gamma = 10$, and $N = 10^6$ finds the minimum value of the objective function 1.5×10^{-14}. Search time was 13 seconds.

5 Conclusion

The results obtained show that the presented very simple homogeneous random search algorithm is quite effective. The presented random search can be successfully used to solve optimization problems. The algorithm itself is easy to use, and choosing search parameters is not a difficult task. At the same time, the program allows solving problems with the utmost accuracy that can be obtained using the double number format of the programming language C#.

References

[1] Ermakov S M and Zhiglyavsky A A 1983 On a random search for a global extremum *Probability Theory and its Applications* **1** 129–136

[2] Ermakov S M, Zhiglyavsky A A and Kondratovich M V 1989 Comparison of some procedures for random search of a global extremum *Journal of Computational Mathematics and Mathematical Physics* **Vol 29 2** 163–170

[3] Zhigljavsky A and Zilinskas A 2008 *Stochastic global optimization* (Berlin: Springer-Verlag)

[4] Zhigljavsky A and Zilinskas A 2016 Stochastic global optimization: a review on the occasion of 25 years of Informatica *Informatica* **Vol 27 2** 229–256

[5] Spall J C 2003 *Introduction to stochastic search and optimization: estimation, simulation, and control* (New Jersey: Wiley)

[6] Ingber L 1989 Very fast simulated re-annealing *Mathl. Comput. Modelling* **Vol 12** 967–973

[7] Lopatin A S 2005 Annealing method, Stochastic optimization in computer science **Vol. 1** 133–149

[8] Granichin O N and Polyak B T 2003 Randomized estimation and optimization algorithms with almost arbitrary noise (Moscow: Nauka)

[9] Sushkov Yu A 1972 *On one method of organizing a random search, Research of operations and statistics modeling* (Leningrad: Publishing House of Leningrad State University) **1** 180–186

[10] Nekrutkin V V and Tikhomirov A S 1993 Speed of convergence as a function of given accuracy for random search methods *Acta Applicandae Mathematicae* **33** 89–108

[11] Tikhomirov A S 2006 On the Markov homogeneous optimization method *Computational Mathematics and Mathematical Physics* **Vol. 46 3** 361–375

[12] Tikhomirov A S 2007 On the convergence rate of the Markov homogeneous monotone optimization method *Computational Mathematics and Mathematical Physics* **Vol. 47 5** 780–790

[13] Tikhomirov A, Stojunina T and Nekrutkin V 2007 Monotonous random search on a torus: integral upper bounds for the complexity *Journal of Statistical Planning and Inference* **Vol. 137 12** 4031–4047

[14] Tikhomirov A S 2020 On the program implementation of a simple Markov homogeneous random search algorithm of an extremum *Journal of Physics: Conference Series* **Vol. 1658 012058** 1–8.