Comparison of dimensionality reduction schemes for parallel global optimization algorithms*

Konstantin Barkalov, Vladislav Sovrasov, Ilya Lebedev, and

Lobachevsky State University of Nizhny Novgorod, Nizhny Novgorod, Russia
konstantin.barkalov@itmm.unn.ru
sovrasov.vlad@gmail.com
ilya.lebedev@itmm.unn.ru
http://hpc-education.unn.ru/основные-напраления/глобальная-оптимизация

Аннотация This work considers a parallel algorithms for solving multiextremal optimization problems. Algorithms are developed within the framework of the information-statistical approach and implemented in a parallel solver "Globalizer" . The optimization problem is solved by reducing the multidimensional problem to a set of joint one-dimensional problems that are solved in parallel. Five types of Peano-type space-filling curves are employed to reduce dimension. The results of computational experiments carried out on several hundred test problems are discussed.

Keywords: Global optimization · Dimension reduction · Parallel algorithms · Multidimensional multiextremal optimization · Global search algorithms · Parallel computations

1 Introduction

Global (or multiextremal) optimization problems are among the most complex problems in both theory and practice of optimal decision making. In these kinds of problems, the optimization criterion can have several local optima within the search domain. The existence of several local optima makes finding the global optimum difficult essentially, since it requires examining the whole feasible search domain. The volume of computations for solving global optimization problems can increase exponentially with increasing number of varied parameters.

These global optimization problem features impose special requirements on the quality of the optimization methods and on the software to implement these ones. The global optimization methods should be highly efficient, and the software systems should be developed on a good professional basis. In general, the global optimization problems can be solved at a reasonable time by employing parallel computations on modern supercomputing systems only.

* [Pleaseinsert\PrerenderUnicode{РЎ}intopreamble][Pleaseinsert\PrerenderUnicode{СЌ}intopreamble][Pleaseinsert\
[Pleaseinsert\PrerenderUnicode{Р»}intopreamble][Pleaseinsert\PrerenderUnicode{СЃ}intopreamble][Pleaseinsert\P
[Pleaseinsert\PrerenderUnicode{Р»}intopreamble][Pleaseinsert\PrerenderUnicode{Р«}intopreamble][Pleaseinsert\P
[Pleaseinsert\PrerenderUnicode{Рґ}intopreamble][Pleaseinsert\PrerenderUnicode{Р№}intopreamble][Pleaseinsert\P
-...

The general state of the art in the field of global optimization is presented in a number of key monographs [13], [25], [33], [38], [51], [52], [54]. The development of optimization methods, which use the high-performance computational systems to solve the time-consuming global optimization problems, is an area of intensive research — see, for instance, [7], [8], [34], [50], [51]. The obtained theoretical results provide the efficient solutions of many applied global optimization problems in various fields of scientific and technical applications [10], [11], [12], [28], [29], [33], [34], [37], [38].

At the same time, the practical implementation of these global optimization algorithms within the framework of industrial software systems is quite limited. In many cases, software implementations are experimental in nature and are used by the developers themselves to obtain the results from the computational experiments required for the scientific publications. This situation originates from high development costs of the professional software systems, which can be used by numerous users. In addition, the global optimization problems could be solved in an automatic mode rarely because of the complexity of these ones. The user should actively control the global search process that implies an adequate level of qualification in the field of optimization (particularly, the user should know and understand the global optimization methods well).

In this work, the authors consider an approach to minimizing multiextremal functions developed in ...???????? This allows problems to be solved in which function values may not be deter-mined for the entire search domain. Under this approach, solving multidimen-sional problems is reduced (using Peano-type space-filling curves) to solving equivalent one-dimensional problems. It should be noted that standard approaches to algorithm parallelization are not quite applicable to global optimization. For example, the rules for selecting an-other iteration point are quite simple and do not require parallelization (as over-heads associated with organizing parallel computations will nullify any possible acceleration). Some acceleration can be achieved by parallelizing the computa-tion of function values describing the object to be optimized; however, this ap-proach is specific to each individual problem being solved. The following approach looks more promising. The algorithm can be modified to run several trials in parallel. This approach provides the efficiency (as paral-lelization is applied to the most computation-intensive part of the problem solv-ing process) and generality (in that it applies to a wide range of global optimiza-tion algorithms). The approach, described in [55] for unconstrained optimization, was used in this work for parallelizing constrained optimization algorithms.

2 Statement of Multidimensional Global Optimization Problem

In this paper, the core class of optimization problems, which can be solved using Globalizer, is formulated. This class involves the multidimensional global optimization problems without constraints, which can be defined in the following way:

$$\varphi(y^*) = \min\{\varphi(y) : y \in D\}, \qquad (1)$$
$$D = \{y \in \mathbf{R}^N : a_i \le y_i \le b_i, 1 \le i \le N\}$$

with the given boundary vectors a and b. It is supposed, that the objective function $\varphi(y)$ satisfies the Lipschitz condition

$$|\varphi(y_1) - \varphi(y_2)| \le L\|y_1 - y_2\|, y_1, y_2 \in D, \qquad (2)$$

where $L > 0$ is the Lipschitz constant, and $||\cdot||$ denotes the norm in $\mathbf{R}^N$ space.

Usually, the minimized function $\varphi(y)$ is defined as a computational procedure, according to which the value $\varphi(y)$ can be calculated for any vector $y \in D$ (let us further call such a calculation *a trial*). It is supposed that this procedure is a time-consuming one. As a result, the overall time of solving the optimization problem (1) is determined, first of all by the number of executed trials. It should also be noted that the requirement of the Lipschitz condition (2) is highly important, since an estimate of the global minimum can be constructed on the basis of a finite number of computed values of the optimized function only in this case .

As it has been shown earlier by many researchers (see, for instance, [12], [25], [38], [51]), finding the numerical estimate of the global optimum implies constructing a coverage of the search domain D. As a result, the computational costs of solving the global optimization problems are readily very high even for a small number of varied parameters (the dimensionality of the problem). A notable reduction in the volume of computations can be achieved when the coverage of the search domain is non-uniform, i. e. the series of trial points is only dense in a vicinity of the global optimum point. The construction of such a non-uniform coverage could be provided in an adaptive way, when the selection of the next trial points is determined by using the search information (the preceding trial points and the values of the minimized function at these points) obtained in the course of computations. This necessary condition complicates considerably the computational schemes of global optimization methods since it implies a complex analysis of a large amount of multidimensional search information. As a result, many optimization algorithms use various approaches to the dimensional reduction [38], [46], [48], [50], [51].

3 Methods of Dimension Reduction

3.1 Базовый алгоритм

Within the framework of the information-statistical global optimization theory, the Peano space-filling curves (or evolvents) $y(x)$ mapping the interval $[0, 1]$ onto an N-dimensional hypercube D unambiguously are used for the dimensionality reduction [46], [48], [50], [51].

As a result of the reduction, the initial multidimensional global optimization problem (1) is reduced to the following one-dimensional problem:

$$\varphi(y(x^*)) = \min\{\varphi(y(x)) : x \in [0,1]\}. \tag{3}$$

It is important to note that this dimensionality reduction scheme transforms the minimized Lipschitzian function from (1) to the corresponding one-dimensional function $\varphi(y(x))$, which satisfies the uniform Hölder condition, i. e.

$$|\varphi(y(x_1)) - \varphi(y(x_2))| \le H|x_1 - x_2|^{\frac{1}{N}}, x_1, x_2 \in [0,1], \tag{4}$$

where the constant H is defined by the relation $H = 2L\sqrt{N+3}$, L is the Lipschitz constant from (2), and N is the dimensionality of the optimization problem (1).

The algorithms for the numerical construction of the Peano curve approximations are given in [51]. As an illustration, an approximation of the Peano curve for the third density level is shown in Figure 1. Figure 1 demonstrates the movement order in a two-dimensional domain to construct the Peano curve approximation; the precision of the Peano curve approximation is determined by the density level used in the construction.

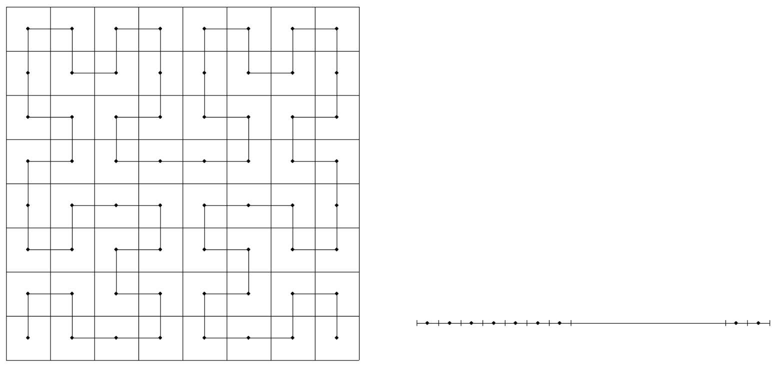

Рис. 1. A Peano curve approximation for the third density level

The computational scheme obtained as a result of the dimensionality reduction consists of the following (see Figure 2):

- The optimization algorithm performs the minimization of the reduced one-dimensional function $\varphi(y(x))$ from (3),
- After determining the next trial point x, a multidimensional image y is calculated by using the mapping $y(x)$,
- The value of the initial multidimensional function $\varphi(y)$ is calculated at the point $y \in D$,
- The calculated value $z = \varphi(y)$ is used further as the value of the reduced one-dimensional function $\varphi(y(x))$ at the point x.

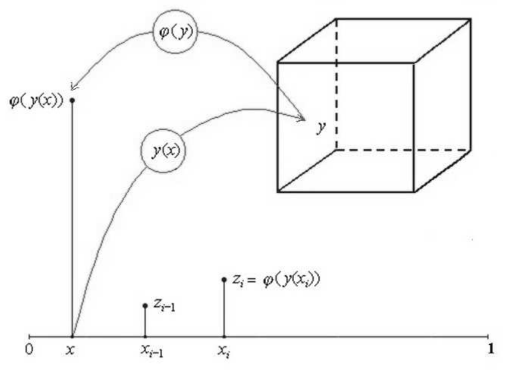

Рис. 2. The computational scheme for obtaining the value of the reduced one-dimensional function $\varphi(y(x))$

3.2 Сдвиговые

s

3.3 Вращаемые

s

3.4 Еще одни развертки

s

3.5 Гладкие

s

4 Parallel Computations for Solving Global Optimization Problems.

4.1 Core Multidimensional Algorithm of Global Search

The information-statistical theory of global optimization formulated in [48], [51] has served as a basis for the development of a large number of efficient multiextremal optimization methods [1], [18], [19], [23], [30], [42], [43], [45], [46], [47].

The optimization methods applied in Globalizer are based on the MAGS method, which can be presented as follows — see [48], [51].

Let us introduce a simpler notation for the optimization problem being solved

$$f(x) = \varphi(y(x)) : x \in [0, 1]. \quad (5)$$

The initial iteration of the algorithm is performed at an arbitrary point $x^1 \in (0,1)$. Then, let us suppose that k, $k \geq 1$, optimization iterations have been completed already. The selection of the trial point x^{k+1} for the next iteration is performed according to the following rules.

Rule 1. Renumber the points of the preceding trials by the lower indices in order of increasing value of coordinates

$$0 = x_0 < x_1 < ... < x_{k+1} = 1, \tag{6}$$

the points x_0, x_{k+1} were introduced additionally for the convenience of further explanation, the values of the minimized function z_0, z_{k+1} at these points are undefined.

Rule 2. Compute the current estimate of the Hölder constant H from (4)

$$M = \max_{1<i\leq k} \frac{|z_i - z_{i-1}|}{\rho_i}, \; m = \begin{cases} rM, \; M > 0, \\ 1, \quad M = 0, \end{cases} \tag{7}$$

as the maximum of the relative differences of the minimized function values on the set of previously executed trial points $x_i, 1 \leq i \leq k$ from (6). Hereafter, $\rho_i = (x_i - x_{i-1})^{\frac{1}{N}}, 1 \leq i \leq k+1$. The constant r, $r > 1$, is the reliability parameter of the algorithm.

Rule 3. Compute the characteristics $R(i)$ for each interval $(x_{i-1}, x_i), 1 \leq i \leq k+1$, where

$$R(i) = 2\rho_i - 4\frac{z_i}{m}, \quad i = 1,$$
$$R(i) = \rho_i + \frac{(z_i - z_{i-1})^2}{m^2\rho_i} - 2\frac{z_i + z_{i-1}}{m}, \quad 1 < i < k+1, \tag{8}$$
$$R(i) = 2\rho_i - 4\frac{z_{i-1}}{m}, \quad i = k+1.$$

Rule 4. Determine the interval with the maximum characteristic

$$R(t) = \max_{1\leq i\leq k+1} R(i). \tag{9}$$

Rule 5. Execute a new trial at the point x^{k+1} located within the interval with the maximum characteristic from (9)

$$x^{k+1} = \frac{x_t + x_{t-1}}{2} - \text{sign}(z_t - z_{t-1})\frac{1}{2r}\left[\frac{r|z_t - z_{t-1}|}{m}\right]^N, \text{ if } 1 < t < k+1, \tag{10}$$

$$x^{k+1} = \frac{x_t + x_{t-1}}{2}, \text{ if } t = 1 \text{ or } t = k+1.$$

The stopping condition, which terminated the trials, is defined by the inequality

$$\rho_t < \varepsilon \tag{11}$$

for the interval with the maximum characteristic from (9) and $\varepsilon > 0$ is the predefined accuracy of the optimization problem solution. If the stopping condition is not satisfied, the index k is incremented by 1, and the new global optimization iteration is executed.

In order to explain the algorithm presented above, let us note the following. The characteristics $R(i), 1 \leq i \leq k+1$, calculated according to (8) could be interpreted as some measures of importance of the intervals with respect to the expected location of the global minimum point. Thus, the rules (9) and (10) for selecting the interval of the next trial become more clear — the point of every next global optimization iteration is selected within the interval, where the global minimum point can be found most likely.

The convergence conditions of the described algorithm are given, for example, in [51].

4.2 Параллельные множественные отображения

The reduction of the multidimensional problems to the one-dimensional ones using evolvents has such important properties as the continuity and preservation of boundedness of function divided differences. However, a partial loss of information on the nearness of the points in the multidimensional space takes place since a point $x \in [0,1]$ has only the left and the right neighbors while the corresponding point $y(x) \in R^N$ has the neighbors in $2N$ directions. As a result, when using the mappings like Peano curve the images y', y'', which are close to each other in the N-dimensional space can correspond to the preimages x', x'', which can be far away from each other in the interval [0,1]. This property results in the excess computations since several limit points x', x'' of the trial sequence generated by the index method in the interval [0,1] can correspond to a single limit point y in the N-dimensional space.

One of the possible ways to overcome this disadvantage consists in using the multiple mapping $Y^S(x) = y^1(x), \ldots, y^S(x)$ instead of single evolvent $y(x)$. To construct the set $Y^S(x)$ different approaches can be used. For example, in [51] a scheme was implemented, according to which each evolvent $y^i(x)$ from $Y^S(x)$ is constructed as a result of shifting the original evolvent $y^0(x)$ along the main diagonal of the hypercube D. The set of Peano curves thus constructed allows one to obtain y', y'' from D for any close multidimensional images, which differ only in one coordinate, close preimages x', x'' from the interval [0,1] for the evolvent $y^s(x), 1 \leq s \leq S$. Using the multiple mapping allows solving initial problem (1) by parallel solving the problems

$$\min\{\varphi(y^s(x)) : x \in [0,1]\}, 1 \leq s \leq S$$

on a set of intervals [0,1] by the index method. Each one-dimensional problem is solved on a separate processor. The trial results at the point x^k obtained for the problem being solved by particular processor are interpreted as the results of the trials in the rest problems (in the corresponding points $x^(k_1), \ldots, x^(k_S)$). In this approach, a trial at the point $x^k \in [0,1]$ executed in the framework of the s-th problem, consists in the following sequence of operations.

1. Determine the image $y^k = y^s(x^k)$ for the evolvent $y^s(x)$.

2. Inform the rest of processors about the start of the trial execution at the pointy^k (the blocking of the point y^k).

3. Determine the preimages $x^{k_s} \in [0,1], 1 \leq s \leq S$, of the point y^k and interpret the trial executed at the point $y^k \in D$ as the execution of the trials in the S points $x^{k_1}, \dots, x^{k_s}$

4. Inform the rest of processors about the trial results at the point y^k.

The decision rules for the proposed parallel algorithm, in general, are the same as the rules of the sequential algorithm (except the method of the trial execution). Each processor has its own copy of the software realizing the computations of the problem functions and the decision rule of the index algorithm. For the organization of the interactions among the processors, the queues are created on each processor, where the processors store the information on the executed iterations in the form of the tuples: the processor number s, the trial point x^{k_s}.

The proposed parallelization scheme was implemented with the use of MPI technology. Main features of implementation consist in the following. A separate MPI-process is created for each of S one-dimensional problems being solved, usually, one process per one processor employed. Each process can use p threads, usually one thread per an accessible core.

At every iteration of the method, the process with the index $s, 0 \leq s < S$ performs p trials in parallel at the points $x^(s + iS), 0 \leq i < p$. At that, each process stores all S_p points, and an attribute indicating whether this point is blocked by another process or not is stored for each point. Let us remind that the point is blocked if the process starts the execution of a trial at this point.

At every iteration of the algorithm, operating within the s-th process, determines the coordinates of p «its own» trial points. Then, the interchange of the coordinates of images of the trial points $y^(s + iS), 0 \leq i < p, 0 \leq s < S$ is performed (from each process to each one). After that, the preimages $x^(q + iS), 0 \leq q < S, q \neq s$ of the points received by the s-th process from the neighbor ones are determined with the use of the evolvent $y^s(x)$. The points blocked within the s-th process will correspond to the preimages obtained. Then, each process performs the trials at the non-blocked points, the computations are performed in parallel using OpenMP. The results of the executed trials (the index of the point, the computed values of the problem functions, and the attribute of unblocking of this point) are transferred to all rest processes. All the points are added to the search information database, and the transition to the next iteration is performed.

5 Results of Numerical Experiments

s

5.1 Сравнение последовательных сдвиговых и вращаемых разверток

s

5.2 Параллельные вращаемые развертки

s

5.3 Параллельные вращаемые развертки для задач большой размерности

s

6 Вывод о целесообразности применения того или иного вида разверток для того или иного вида задач

s

7 First Section

7.1 A Subsection Sample

Please note that the first paragraph of a section or subsection is not indented. The first paragraph that follows a table, figure, equation etc. does not need an indent, either.

Subsequent paragraphs, however, are indented.

Sample Heading (Third Level) Only two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

Sample Heading (Fourth Level) The contribution should contain no more than four levels of headings. Table 1 gives a summary of all heading levels.

Таблица 1. Table captions should be placed above the tables.

Heading level	Example	Font size and style
Title (centered)	**Lecture Notes**	14 point, bold
1st-level heading	**1 Introduction**	12 point, bold
2nd-level heading	**2.1 Printing Area**	10 point, bold
3rd-level heading	**Run-in Heading in Bold.** Text follows	10 point, bold
4th-level heading	*Lowest Level Heading.* Text follows	10 point, italic

Displayed equations are centered and set on a separate line.

$$x + y = z \tag{12}$$

Please try to avoid rasterized images for line-art diagrams and schemas. Whenever possible, use vector graphics instead (see Fig. 3).

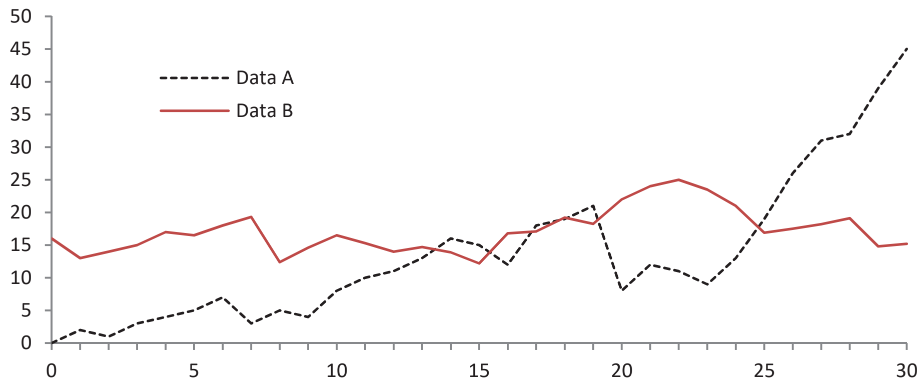

Рис. 3. A figure caption is always placed below the illustration. Please note that short captions are centered, while long ones are justified by the macro package automatically.

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

Доказательство. Proofs, examples, and remarks have the initial word in italics, while the following text appears in normal font.

For citations of references, we prefer the use of square brackets and consecutive numbers. Citations using labels or the author/year convention are also acceptable. The following bibliography provides a sample reference list with entries for journal articles [56], an LNCS chapter [57], a book [58], proceedings without editors [59], and a homepage [60]. Multiple citations are grouped [56,57,58], [56,58,59,60].

Список литературы

1. K. Barkalov and V. Gergel, Multilevel scheme of dimensionality reduction for parallel global search algorithms, in *Proceedings of the 1st International Conference on Engineering and Applied Sciences Optimization*, (2014), 2111–2124.
2. K. Barkalov and V. Gergel, Parallel global optimization on GPU, *J. Glob. Optim.*, **66** (2016), 3–20.
3. K. Barkalov, V. Gergel and I. Lebedev, Use of Xeon Phi coprocessor for solving global optimization problems, *LNCS*, **9251** (2015), 307–318.
4. K. Barkalov, V. Gergel, I. Lebedev and A. Sysoev, Solving the global optimization problems on heterogeneous cluster systems, in *CEUR Workshop Proceedings*, **1482** (2015), 411–419.
5. K. Barkalov, A. Polovinkin, I. Meyerov, S. Sidorov and N. Zolotykh, SVM regression parameters optimization using parallel global search algorithm, *LNCS*, **7979** (2013), 154–166.

6. (MR3618583) M. R. Bussieck and A. Meeraus, General algebraic modeling system (GAMS), in *Modeling Languages in Mathematical Optimization* (ed. J. Kallrath), Springer, (2004), 137–157.
7. Y. Censor and S. A. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*, Oxford University Press, 1998.
8. (MR2499546) R. Čiegis, D. Henty, B. Kågström and J. Žilinskas, *Parallel Scientific Computing and Optimization: Advances and Applications*, Springer, 2009.
9. I. N. Egorov, G. V. Kretinin, I. A. Leshchenko and S. V. Kuptzov, IOSO optimization toolkit — novel software to create better design, in *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2002. Available from http://www.iosotech.com/text/2002_4329.pdf.
10. D. Famularo, P. Pugliese and Y. D. Sergeyev, A global optimization technique for checking parametric robustness, *Automatica*, **35** (1999), 1605–1611.
11. G. Fasano and J. D. Pintér, *Modeling and Optimization in Space Engineering*, Springer, 2013.
12. C. A. Floudas and M. P. Pardalos, *State of the Art in Global Optimization: Computational Methods and Applications*, Kluwer Academic Publishers, Dordrecht, 1996.
13. C. A. Floudas and M. P. Pardalos, *Recent Advances in Global Optimization*, Princeton University Press, 2016.
14. J. M. Gablonsky and C. T. Kelley, A locally-biased form of the DIRECT algorithm, *J. Glob. Optim.*, **21** (2001), 27–37.
15. M. Gaviano, D. E. Kvasov, D. Lera and Y. D. Sergeev, Software for generation of classes of test functions with known local and global minima for global optimization, *ACM Trans. Math. Software*, **29** (2003), 469–480.
16. V. Gergel and I. Lebedev, Heterogeneous parallel computations for solving global optimization problems, *Procedia Comput. Sci.*, **66** (2015), 53–62.
17. V. Gergel, A software system for multi-extremal optimization, *Eur. J. Oper. Res.*, **65** (1993), 305–313.
18. V. Gergel, A method for using derivatives in the minimization of multiextremum functions, *Comput. Math. Math. Phys.*, **36** (1996), 729–742.
19. V. Gergel, A global optimization algorithm for multivariate functions with Lipschitzian first derivatives, *J. Glob. Optim.*, **10** (1997), 257–281.
20. V. Gergel, et al., High performance computing in biomedical applications, *Procedia Computer Science*, **18** (2013), 10–19.
21. V. Gergel, et al., Recognition of surface defects of cold-rolling sheets based on method of localities, *International Review of Automatic Control,* **8** (2015), 51–55.
22. V. Gergel and S. Sidorov, A two-level parallel global search algorithm for solving computationally intensive multi-extremal optimization problems, *LNCS*, **9251** (2015), 505–515.
23. V. A. Grishagin and R. G. Strongin, Optimization of multi-extremal functions subject to monotonically unimodal constraints, *Engineering Cybernetics*, **5** (1984), 117–122.
24. K. Holmstrm and M. M. Edvall, The TOMLAB optimization environment, *Modeling Languages in Mathematical Optimization*, Springer, (2004), 369–376.
25. R. Horst and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer-Verlag, Berlin, 1990.
26. (MR1246501) D. R. Jones, C. D. Perttunen and B. E. Stuckman, Lipschitzian optimization without the Lipschitz constant, *J. Optim. Theory Appl.*, **79** (1993), 157–181.

27. R. B. Kearfott, GlobSol user guide, *Optim. Methods Softw.*, **24** (2009), 687–708.
28. D. E. Kvasov and Y. D. Sergeyev, Deterministic approaches for solving practical black-box global optimization problems, *Adv. Eng. Softw.*, **80** (2015), 58–66.
29. D. E. Kvasov, D. Menniti, A. Pinnarelli, Y. D. Sergeyev and N. Sorrentino, Tuning fuzzy power-system stabilizers in multi-machine systems by global optimization algorithms based on efficient domain partitions, *Electric Power Systems Research*, **78** (2008), 1217–1229.
30. D. E. Kvasov, C. Pizzuti and Y. D. Sergeyev, Local tuning and partition strategies for diagonal GO methods, *Numerische Mathematik*, **94** (2003), 93–106.
31. L. Liberti, Writing global optimization software, in *Nonconvex Optimization and Its Applications*, Springer, **84** (2006), 211–262.
32. Y. Lin and L. Schrage, The global solver in the LINDO API, *Optim. Methods Softw.*, **24** (2009), 657–668.
33. M. Locatelli and F. Schoen, *Global Optimization: Theory, Algorithms and Applications*, SIAM, 2013.
34. G. Luque and E. Alba, *Parallel Genetic Algorithms. Theory and Real World Applications*, Springer-Verlag, Berlin, 2011.
35. M. Mongeau, H. Karsenty, V. Rouzé and J. B. Hiriart-Urruty, Comparison of public-domain software for black box global optimization, *Optim. Methods Softw.*, **13** (2000), 203–226.
36. K. M. Mullen, Continuous global optimization in R, *J. Stat. Softw.*, **60** (2014).
37. M. P. Pardalos, A. A. Zhigljavsky and J. Žilinskas, *Advances in Stochastic and Deterministic Global Optimization*, Springer, 2016.
38. J. D. Pintér, *Global Optimization in Action (Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications)*, Kluwer Academic Publishers, Dordrecht, 1996.
39. J. D. Pintér, Software development for global optimization, *Lectures on Global Optimization. Fields Institute Communications*, **55** (2009), 183–204.
40. L. M. Rios and N. V. Sahinidis, Derivative-free optimization: a review of algorithms and comparison of software implementations, *J. Glob. Optim.*, **56** (2013), 1247–1293.
41. N. V. Sahinidis, BARON: A general purpose global optimization software package, *J. Glob. Optim.*, **8** (1996), 201–205.
42. Y. D. Sergeyev, An information global optimization algorithm with local tuning, *SIAM J. Optim.*, **5** (1995), 858–870.
43. Y. D. Sergeyev, Multidimensional global optimization using the first derivatives, *Comput. Math. Math. Phys.*, **39** (1999), 743–752.
44. Y. D. Sergeyev and D. E. Kvasov, Global search based on efficient diagonal partitions and a set of Lipschitz constants, *SIAM Journal on Optimization*, **16** (2006), 910–937.
45. Y. D. Sergeyev and V. A. Grishagin, Parallel asynchronous global search and the nested optimization scheme, *J. Comput. Anal. Appl.*, **3** (2001), 123–145.
46. Y. D. Sergeyev, R. G. Strongin and D. Lera, *Introduction to Global Optimization Exploiting Space-filling Curves*, Springer, 2013.
47. Y. D. Sergeyev, D. Famularo and P. Pugliese, Index branch-and-bound algorithm for Lipschitz univariate global optimization with multiextremal constraints, *J. Glob. Optim.*, **21** (2001), 317–341.
48. R. G. Strongin, *Numerical Methods in Multi-Extremal Problems (Information-Statistical Algorithms)*, Moscow: Nauka, In Russian, 1978.

49. R. G. Strongin, Algorithms for multi-extremal mathematical programming problems employing a set of joint space-filling curves, *J. Glob. Optim.*, **2** (1992), 357–378.
50. R. G. Strongin, V. P. Gergel, V. A. Grishagin and K. A. Barkalov, *Parallel Computations for Global Optimization Problems*, Moscow State University (In Russian), Moscow, 2013.
51. 5 (MR1797058) [10.1007/978-1-4615-4677-1] R. G. Strongin and Y. D. Sergeyev, *Global Optimization with Non-convex Constraints. Sequential and Parallel Algorithms*, Kluwer Academic Publishers, Dordrecht (2000, 2nd ed. 2013, 3rd ed. 2014).
52. A. Törn and A. Žilinskas, *Global Optimization*, Springer, 1989.
53. P. Venkataraman, *Applied Optimization with MATLAB Programming*, John Wiley & Sons, 2009.
54. A. A. Zhigljavsky, *Theory of Global Random Search*, Kluwer Academic Publishers, Dordrecht, 1991.
55. Gergel, V., Sidorov, S.: A Two-Level Parallel Global Search Algorithm for Solution of Computationally Intensive Multiextremal Optimization Problems. In: Malyshkin, V. (Ed.) PaCT 2015, LNCS, vol. 9251, pp. 505-515. Springer, Heidelberg (2015)
56. Author, F.: Article title. Journal **2**(5), 99–110 (2016)
57. Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016). https://doi.org/10.10007/1234567890
58. Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999)
59. Author, A.-B.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010)
60. LNCS Homepage, http://www.springer.com/lncs. Last accessed 4 Oct 2017