Comparison of several sequential and parallel derivative-free global optimization algorithms

Vladislav Sovrasov[1](✉) and Semen Bevzuk[1]

Lobachevsky State University of Nizhni Novgorod, Russia
sovrasov.vlad@gmail.com, semen.bevzuk@gmail.com

Abstract. This work considers several stochastic and deterministic derivative-free global optimization algorithms. In the first part of the paper popular sequential open-source solvers are compared against the Globalizer solver, which is developed at the Lobachevsky State University. The Globalizer is designed to solve problems with black-box objectives satisfying the Lipschitz condition and shows competitive performance with other similar solvers. The comparison is done on several sets of challenging multi-extremal benchmark functions. The second part of this work is devoted to comparison between the Globalizer and MIDACO solvers on systems with shared and distributed memory. MIDACO is a state-of-the-art global solver included to the TOMLAB optimization environment for MATLAB. Results of the benchmark show advantages of the Globalizer on small-dimensional, but sufficiently multi-extremal benchmark functions.

Keywords: deterministic global optimization · stochastic global optimization · parallel numerical methods · derivative-free algorithms · black-box optimization

1 Introduction

2 Related Work

3 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:

$$\begin{gathered}\varphi(y^*) = \min\{\varphi(y) : y \in D\}, \\ D = \{y \in \mathbb{R}^N : a_i \le y_i \le b_i, 1 \le i \le N\}\end{gathered} \tag{1}$$

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

$$|\varphi(y_1) - \varphi(y_2)| \le L\|y_1 - y_2\|, y_1, y_2 \in D, \tag{2}$$

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

Usually, the objective 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 time-consuming.

4 Dimension Reduction with Evolvents

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 [**?**], [**?**], [**?**], [**?**].

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 Lipschitzian function from (1) to the corresponding one-dimensional function $\varphi(y(x))$, which satisfies the uniform Hölder condition, i. e.

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

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 [**?**].

The computational scheme obtained as a result of the dimensionality reduction consists of the following:

- 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.

4.1 Rotated Evolvents

To overcome complexity of the S-evolvent and to preserve the information on the nearness of the points in the N-dimensional space, one more scheme of building of the multiple mappings was proposed. The building of a set of Peano curves not by the shift along the main diagonal of the hypercube but by rotation of the

evolvents around the coordinate origin is a distinctive feature of the proposed scheme [**?**]. In Fig. **??** two evolvents being the approximations to Peano curves for the case $N = 2$ are presented as an illustration. Taking into account the initial mapping, one can conclude that current implementation of the method allows to build up to $N(N-1)+1$ evolvents for mapping the N-dimensional domain onto the corresponding one-dimensional intervals. Moreover, the additional constraint $g_0(y) \leq 0$ with $g_0(y)$ from (**??**), which arises in shifted evolvents, is absent. This method for building a set of mappings can be "scaled" easily to obtain more evolvents (up to 2^N) if necessary.

5 Sequential and Parallel Algorithm of Global Search

The optimization methods applied in Globalizer to solve the reduced problem (3) are based on the MAGS method, which can be presented as follows — see [**?**], [**?**].

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 $0 = x_0 < x_1 < ... < x_{k+1} = 1$.

Rule 2. Compute the characteristics $R(i)$ for each interval $(x_{i-1}, x_i), 1 \leq i \leq k+1$.

Rule 4. Determine the interval with the maximum characteristic $R(t) = \max_{1 \leq i \leq k+1} R(i)$.

Rule 5. Execute a new trial at the point x^{k+1} located within the interval with the maximum characteristic from the previous step $x^{k+1} = d(x_t)$.

The stopping condition, which terminated the trials, is defined by the inequality $\rho_t < \varepsilon$ for the interval with the maximum characteristic from Step 4 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.

The convergence conditions and exact formulas for descision rules $R(i)$ and $d(x)$ of the described algorithm are given, for example, in [**?**].

5.1 Parallel Algorithm Exploiting a Set of Evolvents

Using the multiple mapping allows solving initial problem (1) by parallel solving the problems

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

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 point y^k (the blocking of the point y^k).

3. Determine the preimages $x^{k_s} \in [0,1], 1 \leqslant s \leqslant S$, 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 $x^{k_1}, \ldots, x^{k_s}$

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

The decision rules for the mentioned 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 mentioned parallelization scheme is implemented in the Globalizer system 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 to parallel execute p trials, usually one thread per an accessible core.

6 Review of Other Methods

6.1 Sequential Methods

- Algorithm of global search (AGSl) (https://github.com/sovrasov/ags_nlp_solver)
- Multi Level Single Linkage (MLSL) (https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#mlsl-multi-level-single-linkage)
- DIRECT (https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#direct-and-direct-l)
- Locally-based DIRECT (DIRECT*l*) (https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#direct-and-direct-l)
- Dual Simulated Annealing (https://github.com/sgubianpm/sdaopt)
- Differential Evolution (https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.differential_evolution.html#scipy.optimize.differential_evolution)
- Controlled Random Search (https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#controlled-random-search-crs-with-local-mutation)
- Simple (https://github.com/chrisstroemel/Simple)
- StoGO (https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#stogo)

6.2 MIDACO Parallel Solver

7 Tools for Comparison of Global Optimization Algorithms

8 Results of Comparison

9 Conclusions

Acknowledgements

The study was supported by the Russian Science Foundation, project No 16-11-10150

References