Solar method for non-convex problems: hypothesizing approach to an optimization

Dmitry Pasechnyuk[1,2*] and Alexander Gornov[3*]

[1]Moscow Institute of Physics and Technology, Dolgoprudny, Russia.
[2]Research Center for Trusted Artificial Intelligence, ISP RAS, Moscow, Russia.
[3]Matrosov Institute for System Dynamics and Control Theory, Irkutsk, Russia.

*Corresponding author(s). E-mail(s): pasechnyuk2004@gmail.com; gornov@icc.ru;

Abstract

This paper is devoted to a new method for non-convex unimodal optimization named *Solar Method*. It consists of subsequential solving some auxiliary optimization problems on randomly chosen low-dimensional sections of the original space. We propose the general scheme of *Solar-like* methods and describe some options for the random generation of secant directions (*rays*) using or not the gradient direction. We investigate the ways to scale the proposed scheme by several levels of linear dependence on the chosen basic variables. We consider several possible approaches to optimization on subspaces: various settings of the dimension of problems, local and global methods to solve them. We compare the operation of the Solar Method, Steepest Descent method, and Conjugate Gradient Method on Rosenbrock–Skokov function in numerical experiments that demonstrate the competitiveness of the proposed approach.

Keywords: Secant subspaces, Optimization on random subspace, Q-search method, Low-dimensional optimization

1 Introduction

Exploring the fundamentally new constructive means to efficiently solve extremal problems belonging to a class of unimodal optimization is an urgent and essentially open challenge. Striving to achieve the results in this direction, one should experimentally test the potential of many plausible techniques on various problems. Of course, if many experimental results confirm the efficiency and competitiveness of a new scheme, it is reasonable to move on to the theoretical justification of the new method and spend efforts on providing guarantees for it. Again: if at the moment there are enough computational facts that show its practical efficiency. In anticipation of this, it is also reasonable to submit some first observations on the algorithm's operation to let the discussion by the community of practice. A strange example of not following this order is the Two-dimensional Gradient Searching method invented by Yu.E. Nesterov more than thirty years ago (personal correspondence), the theoretical design of which is carried out quite recently [1].

In this paper, we look to the concept of two-level methods. One traditionally uses such algorithms in two-level operations research problems. We propose to apply a similar scheme for standard, single-level mathematical programming problems. The approach is received the conventional name *Solar Method*. The main idea of the approach is a step-by-step non-deterministic decomposition of the set of variables into dependent and independent ones. The dependent variables vary within the random rays, going through the "record" point and parameterized by the independent variables (hence the method's name). Thus, the trick here is the two-level scheme, the one level of which is determined by the variables of the second one. At each step of the algorithm, the original problem is replaced by an auxiliary low-dimensionality problem on the chosen ray. We consider various methods for solving the optimization problem with respect to independent variables, including both local and non-local search algorithms. In particular, we used the well-known Nelder–Mead method and the Q-search method we are developing [2].

Further, we denote the uniform (categorical, if discrete) distribution on the set of values X by $\mathcal{U}X$, the special case is $\mathcal{U}\{x_i\} := \mathcal{U}\{x_i\}_{i=1}^n$, where $\{x_i\}_{i=1}^n := \{x_1, x_2, ..., x_n\}$, denote the i-th coordinate of the vector by $[x]_i$, denote the partial per-coordinate derivative by $\nabla_i f(x) := \frac{\partial f(x)}{\partial [x]_i}$. In the algorithms, $x \sim \mathcal{U}$ means that random variable x is drawn from the distribution $\mathcal{U}$, "=" after the variable means the declaration with definition, while "$\leftarrow$" means only a re-definition of the previously declared variable.

The paper is structured as follows. In Section 2 we introduce the concept of *ray* that is the generalization of the subspace notion in a framework of the proposed hypothesizing optimization scheme. We also provide a formal description of some possible algorithms generating rays. In Section 3 we formally describe the proposed optimization method itself. In Section 4 we explore the options of methods to solve the auxiliary problems, discuss some low-dimensional methods suitable for the case of the small dimension of rays,

that is recommended regime for the proposed approach. Here we also present some numerical experiments demonstrating the specific details of the method's operation and advantages in comparison with the Gradient Method and Conjugate Gradient Method. In Section 5 we discuss the scaling of the proposed scheme allowing to complicate method if needed.

2 Random secant subspaces and *rays*

Before describing the scheme of the Solar Method, let us introduce the concept of secant subspaces and, more generally, rays. Further, we will define the auxiliary problems within the Solar Method as contractions of the original problem into them. At the same time, the choice of parameters (directions) of these rays is a wide enough question to be taken out in a separate section of the paper. The proposed approach is based on the following method of forming the ray:

1. choosing coordinates defining the direction, the basis of the subspace,
2. splitting remaining coordinates of original space into groups, binding them with base coordinates (one original coordinate is dependent on one base coordinate),
3. choosing the parameters of these dependencies.

Now, we describe formally the algorithm implementing this sequencing:

Algorithm 1 Procedure generating random rays (with linear dependencies)

Require: n is a dimensionality of original space, x is a point in original space, b is a number of base coordinates (dimensionality of subspace), A_k is an amplitude of the proportionality coefficients for linear dependencies

1: $b_j \sim \mathcal{U}\{1, ..., n\}\ \forall j \in \{1, ..., b\}$
2: **for** $i \in \{1, ..., n\} \setminus \{b_j\}_{j=1}^{b}$ **do**
3: $\quad k_i \sim \mathcal{U}[-A_k, A_k]$
4: $\quad h_i = x_i - k_i \cdot x_{b_m}, b_m \sim \mathcal{U}\{b_j\}$
5: **end for**

There is already come out a substantial drawback of the outlined scheme. Indeed, within this variety of base directions and parameters for linear dependencies, we cannot guarantee that randomly chosen parameters provide a suitable relaxation for considered function every time. Even for a simple convex function (quadratic, for instance), one can choose a subspace close enough to the tangent for some contour line, for which the decrease of function value at the method step would be minimal, with whatever accuracy the auxiliary problem is solved. Moreover, for the arbitrary unimodal function, some chosen subspace may provide a fruitful relaxation to a problem (in terms of decrease in function value), but the next point obtained with it, corresponding to the solution of the corresponding auxiliary problem, may be sent in the direction

opposite to the direction to a global minimum. It can lead to a divergence in the argument and an expected slowdown in function value decrease.

The method we propose solves the problem above with a conceptually new approach. To reveal it, let us look at the problem statement we have from such a detached point of view: in line with the classic oracle approach to optimization [3], we have no information about the function we optimize at the beginning, while our goal is to collect enough information to find its minimum. Further, inspired by the general natural scientific approach, we can formulate any (unwarranted) hypotheses (of some specific form) and test them to explore the problem instead of using a standard gradient oracle. If the hypotheses are simple enough, it is easy to iterate over them (generate them randomly) until the next one sufficiently simplifies a problem. For example, in Algorithm 1 we assume that the change in function is determined by some base variables, while other variables can be set linearly dependent without loss of information. The judging on the quality of a hypothesis is based on the assessment of the corresponding relaxation quality. Of course, some of them may turn out to be unproductive (most of them will), and in this case, it would be reasonable to discard them, waiting for a more successful generation. In the method we propose, it is expressed with such a concise verification:

if $f(\widetilde{x}_{t+1}) < f(x_t)$ **then**
  $x_{t+1} \leftarrow \widetilde{x}_{t+1}$;
end

where $\widetilde{x}_{t+1}$ is a (approximate) solution for the auxiliary problem corresponding to a ray.

Thus, the approach we propose relies on generating many candidate rays (bases and dependencies parameters), some of which become discarded, while the others are confirmed and accepted to form the next point of the method.

Nevertheless, the question of generating more suitable and less poor rays remains a key. Even for the simple linear parametrization, we have an impressive cardinality of possible rays parameters, and we need to reduce this set or generate more favorable random configurations more likely. A natural way to achieve this is to use the available first-order information to form new rays at every iteration (point) of the method. More precisely, we can generate rays with a direction close to that one for the gradient. Indeed, since (anti-)gradient provides the steepest local decay direction, it is reasonable to assume that:

1. the minimum within the close (in terms of somewhat "angle" between one-dimensional subspace defined by the gradient and the chosen one) subspace would be better than a random one,
2. hypothesis of near-linear behavior of the function would be more likely with parameters defined by gradient, at least in the vicinity of the point.

On the other hand, if the parameters of linear dependencies correspond to the gradient direction exactly, optimization along such a ray will lead to the

same point as for the standard steepest gradient descent step. To preserve the freedom of the hypothesized approach, we take the weighted sum of gradient and a random direction to fit the ray. More formally, we propose the following algorithm for forming the rays:

Algorithm 2 Gradient-based procedure generating random rays

Require: n is a dimensionality of original space, x is a point in original space, b is a number of base coordinates, A_k is an amplitude of the proportionality coefficients for linear dependencies, α is a weighting factor

1: $b_j \sim \mathcal{U}\{1, ..., n\}\ \forall j \in \{1, ..., b\}$
2: **for** $i \in \{1, ..., n\} \setminus \{b_j\}_{j=1}^{b}$ **do**
3: $\quad \widetilde{k}_i \sim \mathcal{U}[-A_k, A_k]$
4: $\quad b_m \sim \mathcal{U}\{b_j\}$
5: $\quad k_i = \alpha \cdot (\nabla_i f(x_i)/\nabla_i f(x_{b_m})) + (1 - \alpha) \cdot \widetilde{k}_i$
6: $\quad h_i = x_i - k_i \cdot x_{b_m}$
7: **end for**

Of course, this does not solve the problem: there remains significant variability in the choice of rays. Alas, in the general class of unimodal functions, most of them can be quite disadvantageous. One way to partially overcome this is to equip the generating algorithm with a supervised learning technique with the aim of more reduce the generation's number. The latter could take the successes and failures of the previously formulated hypotheses to form the next one or prune some possible choices. Note that this concept is independent of the chosen linear form of dependencies and is quite general. In this way, one can apply a whole variety of simple gradient direction correcting policies (momentum, exponential moving average, conjugate gradient-like). One can also use some advanced practices (meta-learning) to explore the function landscape by generating probes. Machine learning is an expected extension of this list of exploration techniques. The breadth of perspectives is immense and is out of the scope of this paper.

Remark 1 The proposed scheme allows using both linear rays (subspaces with a linear dependence of coordinates on the base ones) and non-linear ones with more complicated dependencies (for example, polynomial). In this case, we have two options: generate coefficients for polynomial randomly or form them fitting some probes that are points and gradients (interpolation regime). For example, in a two-dimensional space, one can lead a parabola through the three points, or the two points, if the derivative in one of them is given.

Following this, we can propose a simple modification to a special case of the proposed scheme. Let us name it as Mustache Method. Its iteration consists of randomly generating several probes (in a given box-constrained feasible set), analytically plotting a curve through them, and then applying curvilinear optimization techniques [4] to form the next point of the method. In this method, *mustache* is a one-dimensional ray. One can easily extend this approach to a general Solar Method scheme.

3 Solar Method scheme

Now we return to the discussion of the proposed scheme. To begin with, let us explicitly describe the formulation of the problem being solved:

$$\min_{x \in B} f(x),$$

where f is a continuous unimodal function (if algorithm uses gradient in rays generating procedure, function is also differentiable), $B \subset \mathbb{R}^n$ is a box-constraint subset of the original space, that is $B = [a_1, b_1] \times ... \times [a_n, b_n]$.

In the previous section we introduced the concept of the ray. This was necessary in order to unify the shape of the subsets, the constricted of the original problem to which would be the auxiliary problems. Let us formulate formally their structure:

$$\min_{y \in \widetilde{B}} \varphi(y), \tag{1}$$

where $\widetilde{B} \subset B$ is box-constrained feasible set constricted to the selected base coordinates, and φ is an auxiliary function that is a constriction of f to the chosen ray, parametrized with $\{k_i\}$, $\{h_i\}$, $\{b_m(i)\}$ ($x|_B$ here means a projection of the point onto the specified set by applying the following transformation $x_i = \max\{a_i, \min\{x_i, b_i\}\}$):

$$\varphi(y) := f(x(y)|_B),$$

$$\text{where } x_i(y) := \begin{cases} y_j & \text{if } i \in \{b_j\} \\ h_i + k_i \cdot y_{b_m(i)} & \text{else} \end{cases}, \quad \forall i \in 1, ..., n.$$

Let us move now to the description of the Solar Method: the algorithm combining the rays generating procedure, sorting out the formulated hypotheses, and solving the auxiliary problems. The algorithm is simple: relying on the last point generated by the method, it constructs the ray with random parameters that go through this point, then constricts an original problem on this ray and solves it with some other method. If the solution obtained on the ray and lifted to the full-dimensional space is better than the current point, we accept it as the next point of the method. The searching for a solution on a random linear subspace selected from the entire manifold resembles movement along a ray of the sun, and this analogy gave the name to the method. In following Algorithm 3 we provide a more formal description of the approach explained above:

Let us note some practically interesting properties of this method following directly from its construction and a concept of hypothesizing approach:

1. **monotone**. Each drawn subspace contains at least the reference point through which it goes, and therefore, if we set method $\mathcal{M}$ to the small enough accuracy and get the corresponding solution for the auxiliary problem, it is possible to find a point from the vicinity of the reference or

Algorithm 3 Solar Method

Require: x_0 is a starting point, T is a number of hypothesis generations, b is a number of base coordinates, $\mathcal{M}$ is a method optimizing corresponding b-dimensional functions

for $t = 0, ..., T-1$ **do**
  Generate parameters $\{k_i\}, \{h_i\}, \{b_m(i)\}$ of the random ray through the point x_t
  $y_t = [[x_t]_{b_1}, ..., [x_t]_{b_b}]$
  By running the method $\mathcal{M}$ from the starting point y_t
find the solution $y^\star$ of the auxiliary problem (1)
  $[\widetilde{x}_{t+1}]_i = \begin{cases} y^\star_j & \text{if } i \in \{b_j\} \\ h_i + k_i \cdot y^\star_{b_m(i)} & \text{else} \end{cases}, \quad \forall i \in 1, ..., n$
  if $f(\widetilde{x}_{t+1}) < f(x_t)$ **then**
    $x_{t+1} \leftarrow \widetilde{x}_{t+1}$
  end if
end for

the reference itself as an approximate solution to the auxiliary problem. So, evaluating the function in every next point, we obtain the guaranteed non-increasing sequence of values.

2. **no long-term convergence decline**. The described method is highly stochastic, and its convergence depends on the advantageousness of the rays generated at each iteration. However, no matter how complex the landscape of the optimized function is, it has a minimum point and hence has a subspace going through it or in its vicinity. Thus, by a possible long search, the method will sooner or later choose an advantageous direction and improve the last solution. The method's convergence curve is abrupt due to this.

4 Subsolvers and numerical experiments

An essential hyperparameter of the Solar Method is the auxiliary method $\mathcal{M}$ used to solve the problems on every selected ray. There is great freedom in choosing this method. But due to the usual (and recommended) small number of variables in the basis b and hence the low-dimensionality of auxiliary problems, it is reasonable to choose methods that are effective in solving low-dimensional problems. Here we face a trade-off: to solve every subproblem with high accuracy (to find a point that significantly improves the last one) or solve them very quickly (to have time for checking more hypotheses). Indeed, this is a standard compromise in optimization theory. But note that in the proposed scheme, this question is not a key one. If the accuracy is not enough, we can skip the hypothesis even if it is good, hoping to generate a not worse further. And vice versa, if we get the best out of every generation, we can perform more small steps accumulating progress.

Remark 2 A promising perspective here is to use two methods in the compound: one for assessing the chosen hypothesis and another for the corresponding auxiliary function optimization. Due to the presence of box constraints in the problem statement, we can check the quality of chosen ray by evaluating the function in several uniformly random probes in a given feasible set. A call for the zero-order oracle at some points can be very effective in terms of working time. In this way, one could reduce the time of the method's operation, spent on unsuccessful strategies, by their preliminary filtering.

In this section, we consider some options for the method to solve the auxiliary problems in the Solar Method. The first of them is the widely-known Nelder–Mead method [5], which proved its practical efficiency in many independent numerical experiments [6]. The second considered method is the lesser-known Q-search algorithm proposed by A.Yu. Gornov. Insofar as we cannot provide an appropriate reference to some original paper devoted to the Q-search, we describe the method below. Its iteration consists of generating several random probes in the selected box-constrained feasible set, choosing one of them as a reference (for some rule of choosing), determining the best point among the reference point and the last point generated by the method, and cutting off a part of the box so that the rest of the box has the worst one as its apex and the best one lay in the interior of this new box. Algorithm 4 formally describes the special case of this method (by $[B_{t+1}]_j$ we denote the j-th dimension of the box, that is, a segment $[a_j, b_j]$):

We carried out some numerical experiments to show the operation of differently compared Solar Method in practice. We compare the convergence of the proposed method with the Nelder–Mead method and the Q-search for solving auxiliary problems on the rays and some other classical optimization methods. To test the performance, we use the Rosenbrock–Skokov function [7] (a generalization of the Rosenbrock function to large dimensions) as an example of the complicated non-convex problem revealing some pointed drawbacks of the proposed method. Classical methods we compare with are Steepest Gradient Descent and Conjugate Gradients Method. Gradient Descent is a universally applicable well-known solver and somewhat a baseline one, and Conjugate Gradients Method, in the opposite, established himself in numerous practical problems as a leader. It is a formal problem statement:

$$f(x) = (1 - x_1)^2 + 100 \sum_{i=2}^{n} (x_i - x_{i-1}^2)^2.$$

Figures 1, 2 present the convergence curves for compared methods. The blue line here represents the Conjugate Gradients Method (Fletcher–Reeves version [8]), the orange line represents the Steepest Gradient Descent, the green lines represent the individual curves of the Solar Method (started multiple times), and the blue one is for their averaging. The abscissa axis measures the iterations number, and the ordinate axis measures the function value (in

Algorithm 4 Q-search

Require: y_0 is a starting point, s_{min} is a minimal size of the current box, N is a number of probes, $\widetilde{B}$ is an initial set

$t = 0$
$B_0 = \widetilde{B}$
while $s_{min} < \sum_{j=1}^{b}(\widetilde{b}_j - \widetilde{a}_j)$ **do**
    Generate a set of random probes $\{y^1, ..., y^N\} \subset B_t$
    Choose a probe $y^\star$ with the smallest value of φ from them
    if $f(y^\star) < f(y_t)$ **then**
        $y_{t+1} \leftarrow y^\star$; $y^\star \leftarrow y_t$
    else
        $y_{t+1} \leftarrow y_t$
    end if
    $B_{t+1} \leftarrow B_t$
    for $j = 1, ..., b$ **do**
        if $y_j^\star < [y_t]_j$ **then**
            $[B_{t+1}]_j \leftarrow [B_{t+1}]_j \cap [y_j^\star, +\infty)$
        else
            $[B_{t+1}]_j \leftarrow [B_{t+1}]_j \cap (-\infty, y_j^\star]$
        end if
    end for
    $t \leftarrow t + 1$
end while

logarithmic scale, $\log_{10}$). Figures show that the Solar Method converges on average faster than the Steepest Gradient Descent, but in some runs, it converges faster than the Conjugate Gradient Method thanks to the specific jumps in the function values. Note that the subsolver setting significantly affects the method's behavior: when using the Q-search method, the curve is smoother, and the convergence at the initial iterations of the method slows down. It emphasizes the need to consider various options for subsolvers to find a more appropriate one to use with the proposed method and specific problem.

Remark 3 There are more complicated versions of the Q-search that we can use within the Solar Method. These use more local information (properties of function in the vicinity of the current point) to perform the step. Firstly, we can select the reference probe differently: with maximal/minimal function value, as a center of mass for all generated points (with weights proportional/inversely to the function value). Note that, under certain conditions, we can use parallelization for computations associated with each of the probes to obtain more information about the function during the same operation time. Secondly, we can complicate the Q-search algorithm by using the first-order information: we can apply several rules for cutting the box that would take into account both the relative position of solution and reference points and the gradient.

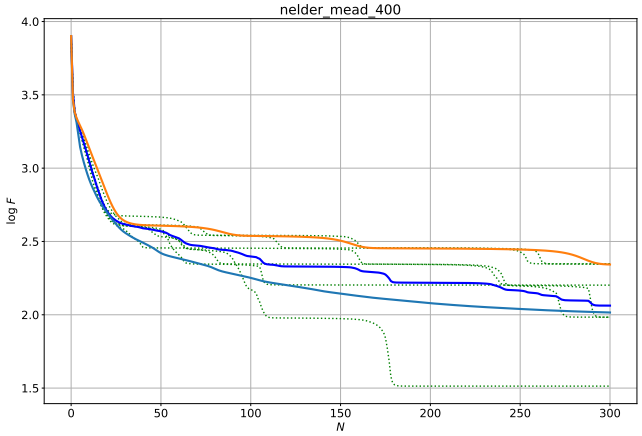

Fig. 1 Convergence curves of the Solar Method with the Nelder–Mead subsolver, $n = 400, b = 15, x \in [-3, 3]^n, \alpha = 0.5$.

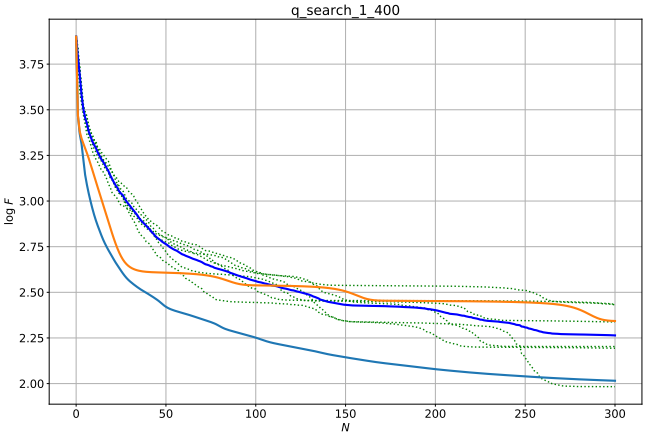

Fig. 2 Convergence curves of the Solar Method with the Q-search subsolver, $n = 400, b = 15, x \in [-3, 3]^n, \alpha = 0.5$.

Remark 4 In many real-life applications, we face not only unimodal but also multi-extrema problems. It is essential for a non-local method like Solar Method to operate efficiently in such a complicated setup. The fundamental barrier here is that method can stick to the local minimum instead of finding the globally optimal solution, and most of the gradient methods developed for convex optimization cannot handle it. The general construction of the Solar Method does not have this drawback.

However, in practice, to apply Solar Method to such problems, it is necessary to use a subsolver avoiding local minima in corresponding low-dimensional cases.

Of course, it is allowed for these methods to form discontinuous trajectories, the descent along which is of a more complex, inconsistent manner and may include conditional switching between the branches of this trajectory. An example is the Subspace Tunneling Method. Its iteration consists of step from the current point in a randomly generated direction with fixed step size, then step in the opposite direction from the same point with the same step size, leading a function through the three obtained probes (using parabolic interpolation, for example) and step to a minimum of this function.

5 Scaling

In this section, we describe some perspectives in one possibly fruitful way to develop the proposed method. It is an extension of the algorithm to multiple levels determined by the dependence of some variables on the base ones. This scaling framework provides a simple, easily manageable way to complicate the method to improve its convergence characteristics. Indeed, such layering allows one to construct more complex hypotheses on the problem, and on the other hand, carefully control the iteration over them and prune some branches of unsuitable options.

From an implementation point of view, this extension means using the Solar Method (or only the rays generating procedure, that is a part of it) as an auxiliary inside another Solar Method. In this way, we construct a hierarchy of dependencies between variables: there a b_1 primarily base variables defined by the outer Solar Method, b_2 of secondary base variables defined by the second level method and dependent on the previous b_1 arguments, and the rest $n - b_2 - b_1$ variables, dependent on the last level's base ones. It is a scheme of 3-level Solar Method, but we can similarly continue this layering.

Mathematically, this means to set the dependence of the final level variables on the primarily base ones in the form of a composition of two or more transformations (depends on the number of levels). Of course, in the case of linear dependencies, no matter how deep the composition is, we obtain the linear transformation again. However, while the composition of linear operators is always a linear operator, here we consider the superpositions of transformations from a more wide class, that is *optimization operators*. Such a view of optimization methods and their transformations, in general, can be fruitful for developing the related theoretical questions.

Let us return now to the specific method we consider in this paper. The practically important note is that for solving some real-life problems, it is often efficient to use optimization schemes reducible to the subproblems with dimensionality equal to 2. Therefore, we additionally list the auxiliary methods applicable to optimize the 2-dimensional version of the original problem in a Solar Method. One of the effective methods in the described setting may be the Yu. E. Nesterov's method for convex optimization on the square [1] acting as a local search within the proposed non-convex optimization scheme. Another option is a modification of the barycentric coordinates method proposed in [9].

6 Conclusion

In this paper, we propose the Solar Method for unimodal optimization problems based on the introduced hypothesizing approach of searching the step direction for the method. We also describe the means to evolve the proposed scheme, such as scaling it to multiple levels or generalization for different rays' forms. The numerical experiments for the Rosenbrock–Skokov function of various dimensions show a fairly regular but jumpwise convergence of the algorithm. Unfortunately, the considered variants of the algorithm are somewhat inferior to the Conjugate Gradient Method but usually outrun the Steepest Descent Method.

References

[1] Pasechnyuk, D.A., Stonyakin, F.S.: One method for minimization a convex lipschitz-continuous function of two variables on a fixed square. Computer Research and Modelling **11**(3), 379–395 (2019)

[2] Gornov, A.Y.: Q-search: a successful zero-order method for unconstrained optimization problems (in russian). In: Proceedings of the Lyapunov Reading ISDCT SB RAS, Irkutsk, Russia

[3] Nemirovskij, A.S., Yudin, D.B.: Problem Complexity and Method Efficiency in Optimization. Wiley-Interscience, New York (1983)

[4] Dennis Jr, J.E., Schnabel, R.B.: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. SIAM, . (1996)

[5] Nelder, J.A., Mead, R.: A simplex method for function minimization. The computer journal **7**(4), 308–313 (1965)

[6] Nelder–Mead Method: Page of the Machinelearning.ru Web-site (in Russian). Is available at http://www.machinelearning.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%9D%D0%B5%D0%BB%D0%B4%D0%B5%D1%80%D0%B0-%D0%9C%D0%B8%D0%B4%D0%B0, date of access to the resource: 24.01.2021

[7] Skokov, V.A.: Methods and Algorithms for Unconstrained Minimization of Functions of Several Variables (review) (in Russian). CEMI USSR Academy of Sciences, Moscow (1974)

[8] Fletcher, R., Reeves, C.M.: Function minimization by conjugate gradients. The computer journal **7**(2), 149–154 (1964)

[9] Kononyuk, A.E.: Fundamentals of Optimization Theory. Unconditional Optimization (in Russian). Education of Ukraine, Kiev (2011)