**КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ
И МОДЕЛИРОВАНИЕ 2022 Т. 14 № 2 С. 239–255**
DOI: 10.20537/2076-7633-2022-14-2-239-255

Ки&М

МАТЕМАТИЧЕСКИЕ ОСНОВЫ И ЧИСЛЕННЫЕ МЕТОДЫ МОДЕЛИРОВАНИЯ

УДК: 519.8

Линейно сходящиеся безградиентные методы для минимизации параболической аппроксимации

А. И. Базарова[1,a], А. Н. Безносиков[1,b], А. В. Гасников[1,2,3,c]

[1]Московский физико-технический институт (национальный исследовательский университет),
Россия, 141701, Московская обл., Долгопрудный, Институтский пер. 9
[2]Институт проблем управления РАН им. А. А. Харкевича,
Россия, 212705, г. Москва, Большой Каретный переулок, д. 19, стр. 1
[3]Кавказский математический центр Адыгейского государственного университета,
Россия, 385000, Республика Адыгея, г. Майкоп, ул. Первомайская, д. 208

E-mail: [a] bazarova.ai@phystech.edu, [b] beznosikov.an@phystech.edu, [c] gasnikov@yandex.ru

Получено 09.02.2022.
Принято к публикации 13.02.2022.

Нахождение глобального минимума невыпуклых функций — одна из ключевых и самых сложных проблем современной оптимизации. В этой работе мы рассматриваем отдельные классы невыпуклых задач, которые имеют четкий и выраженный глобальный минимум.

В первой части статьи мы рассматриваем два класса «хороших» невыпуклых функций, которые могут быть ограничены снизу и сверху параболической функцией. Такой класс задач не исследован широко в литературе, хотя является довольно интересным с прикладной точки зрения. Более того, для таких задач методы первого и более высоких порядков могут быть абсолютно неэффективны при поиске глобального минимума. Это связано с тем, что функция может сильно осциллировать или может быть сильно зашумлена. Поэтому наши новые методы используют информацию только нулевого порядка и основаны на поиске по сетке. Размер и мелкость этой сетки, а значит, и гарантии скорости сходимости и оракульной сложности зависят от «хорошести» задачи. В частности, мы показываем, если функция зажата довольно близкими параболическими функциями, то сложность не зависит от размерности задачи. Мы показываем, что наши новые методы сходятся с линейной скоростью сходимости $\log(1/\varepsilon)$ к глобальному минимуму на кубе.

Во второй части статьи мы рассматриваем задачу невыпуклой оптимизации с другого ракурса. Мы предполагаем, что целевая минимизируемая функция есть сумма выпуклой квадратичной задачи и невыпуклой «шумовой» функции, пропорциональной по модулю расстоянию до глобального решения. Рассмотрение функций с такими предположениями о шуме для методов нулевого порядка является новым в литературе. Для такой задачи мы используем классический безградиентный подход с аппроксимацией градиента через конечную разность. Мы показываем, как можно свести анализ сходимости для нашей задачи к стандартному анализу для задач выпуклой оптимизации. В частности, и для таких задач мы добиваемся линейной скорости сходимости.

Экспериментальные результаты подтверждают работоспособность и практическую применимость всех полученных методов.

Ключевые слова: безградиентная оптимизация, невыпуклая задача, линейная скорость сходимости

Исследование выполнено за счет гранта Российского научного фонда (проект № 21-71-30005).

COMPUTER RESEARCH AND MODELING
2022 VOL. 14 NO. 2 P. 239–255
DOI: 10.20537/2076-7633-2022-14-2-239-255

Ки&М

MATHEMATICAL MODELING AND NUMERICAL SIMULATION

UDC: 519.8

Linearly convergent gradient-free methods for minimization of parabolic approximation

A. I. Bazarova[1,a], A. N. Beznosikov[1,b], A. V. Gasnikov[1,2,3,c]

[1]Moscow Institute of Physics and Technology,
9 Institutskiy per., Dolgoprudny, Moscow region, 141701, Russia
[2]Institute for Information Transmission Problems of the Russian Academy of Sciences (Kharkevich Institute),
19/1 Bol'shoy Karetnyy per., Moscow, 212705, Russia
[3]Caucasus Mathematical Center, Adyghe State University,
208 Pervomaysk st., Maikop, Adyghe, 385000, Russia

E-mail: [a] bazarova.ai@phystech.edu, [b] beznosikov.an@phystech.edu, [c] gasnikov@yandex.ru

Received 09.02.2022.
Accepted for publication 13.02.2022.

Finding the global minimum of a nonconvex function is one of the key and most difficult problems of the modern optimization. In this paper we consider special classes of nonconvex problems which have a clear and distinct global minimum.

In the first part of the paper we consider two classes of «good» nonconvex functions, which can be bounded below and above by a parabolic function. This class of problems has not been widely studied in the literature, although it is rather interesting from an applied point of view. Moreover, for such problems first-order and higher-order methods may be completely ineffective in finding a global minimum. This is due to the fact that the function may oscillate heavily or may be very noisy. Therefore, our new methods use only zero-order information and are based on grid search. The size and fineness of this grid, and hence the guarantee of convergence speed and oracle complexity, depend on the «goodness» of the problem. In particular, we show that if the function is bounded by fairly close parabolic functions, then the complexity is independent of the dimension of the problem. We show that our new methods converge with a linear convergence rate $\log(1/\varepsilon)$ to a global minimum on the cube.

In the second part of the paper, we consider the nonconvex optimization problem from a different angle. We assume that the target minimizing function is the sum of the convex quadratic problem and a nonconvex «noise» function proportional to the distance to the global solution. Considering functions with such noise assumptions for zero-order methods is new in the literature. For such a problem, we use the classical gradient-free approach with gradient approximation through finite differences. We show how the convergence analysis for our problems can be reduced to the standard analysis for convex optimization problems. In particular, we achieve a linear convergence rate for such problems as well.

Experimental results confirm the efficiency and practical applicability of all the obtained methods.

Keywords: zeroth-order optimization, nonconvex problem, linear rate

Citation: *Computer Research and Modeling*, 2022, vol. 14, no. 2, pp. 239–255.
This research was funded by the Russian Science Foundation (project 21-71-30005).

Introduction

Methods for minimizing convex functions are well studied in the literature [Boyd, Vandenberghe, 2018; Nocedal, Wright, 2006] and have good guarantees of convergence to a solution. When the objective function is nonconvex, the problem becomes much more complicated. Meanwhile, the ability to find the global minimum of nonconvex functions is an equally important issue, but in general, this is NP-hard. The main idea of constructing an analysis around nonconvex functions is the introduction of some restrictions on the problem: these can be desire to search not for a global minimum, but only for a local minimum (in the hope that local is good enough) or restrictions on a function on a set of optimization.

In the first part of the paper, we follow the same way and try to find a global minimum of the function bounded by two parabolic functions More formally, our statement of the problem can be described as follows:

$$\min_{x \in C} f(x), \tag{1}$$

where the set C is a cube in $\mathbb{R}^d$, i.e. for all $x \in C$: $l_i \leqslant x_i \leqslant u_i$ with i from 1 to d. We do not know whether $f(x)$ is convex, smooth, whether its gradient is bounded or not. In general, the function can be any function, including nonconvex and nondifferentiable ones. But we assume that the function $f(x)$ satisfies the following condition for all $x \in C$:

$$\frac{\mu}{2}\|x - x^*\|^2 \leqslant f(x) - f(x^*) \leqslant \frac{L}{2}\|x - x^*\|^2. \tag{2}$$

Hereinafter, x^* is the solution to problem (1) and we use the ordinary Euclidean norm $\|\cdot\|$. Inequalities (2) define the "good" class. Such a condition describes a rather large set of functions that has a global minimum on the cube (see Fig. 1).

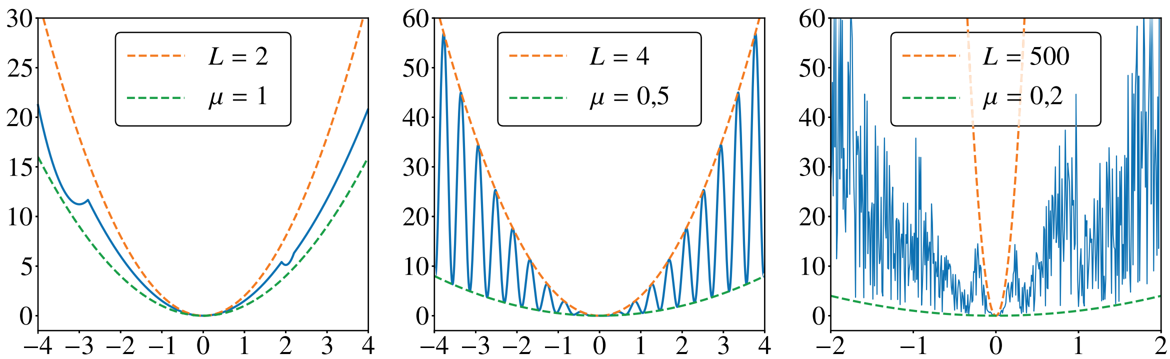

Figure 1. Examples of functions that satisfy condition (2) with different constants L and μ. From left to right, the ratio L/μ increases

One can note that using a first-order oracle (gradient) for such functions is not a good idea. Since, due to possible large and sharp oscillations, the gradient does not carry any useful global information. Thus, the methods outlined in this paper rely exclusively on the zeroth-order oracles.

It seems natural that if the constant L is too large or/and the constant μ is too small, then the search for the solution becomes more difficult. Therefore, we propose another class of "very good" functions for which the constants L and μ differ but not much:

$$f(x) - f(x^*) = \left(\frac{M}{2} + \delta(x)\right)\|x - x^*\|_2^2, \quad \text{with } |\delta(x)| \leqslant \Delta = \frac{M}{16(d-1)}, \tag{3}$$

for all $x \in C$. Such functions are quite quadratic, but they can fluctuate with a level of deviation equal to $\delta(x)$. It is easy to see that condition (2) is satisfied with $L = \frac{M}{2} + \frac{M}{16(d-1)}$ and $\mu = \frac{M}{2} - \frac{M}{16(d-1)}$.

In the second part of the paper we look at the "good" functions described above in different way. Suppose we are given a parabolic function, but the oracle returns not the exact value of this function, but with noise:

$$f(x) = \widetilde{L}\|x - x^*\|^2 + \xi(x) + \delta(x).$$

Here, ξ is responsible for stochastic (random) noise, and δ, for deterministic noise. Moreover, for the functions (2) and (3) $\xi = 0$, and $\delta(x) \sim \|x - x^*\|^2$. In this part of the work we consider a slightly different concept, namely,

$$f(x, \xi) = \frac{1}{2}(x - x^*)^T A(x - x^*) + (\xi + \delta(x))\|x - x^*\|, \tag{4}$$

where $A > 0$. Having only this information about the function, we want to solve the problem:

$$\min_{x \in \mathbb{R}^d} \frac{1}{2}(x - x^*)^T A(x - x^*). \tag{5}$$

For this one can reconstruct the real gradient using finite differences:

$$\frac{d}{2\tau}(f(x + \tau e, \xi^+) - f(x - \tau e, \xi^-))e, \tag{6}$$

where e is some random vector uniformly distributed on the Euclidean sphere. Then such a gradient approximation can be used in Gradient Descent, which it does.

Our contribution and related works

Let us start with a discussion of the ideas of **the first part**.

There are already results in the literature where minimization of some specific class of nonconvex functions is considered. In [Shor, 2012; Polyak, 1987], the objective function is nonconvex, but at the same time, it is bounded from below and above by some "good" functions. Essentially, there is a similar approach in [Singer, Vondrak, 2015], they consider gradient-free minimization of convex functions, but additionally, assume that the zeroth-order oracle takes values of the function with noise. This concept is suitable for nonconvex problems in which the objective function "oscillates" around some convex function.

The idea of our method is remotely similar to the simplest zeroth-order methods for minimizing one-dimensional unimodal functions: we calculate the value of the function at some points, and then, using this information, we decrease the optimization set by a certain number of times.

Also, our methods are partly close to the Monte Carlo type algorithms [Zhigljavsky, Zilinskas, 2007]. These methods are also suitable for nonconvex optimization problems and exploit the idea of Markov search for a solution. However, they also require the problem to be "good" enough. Our methods also do some kind of search, which is not stochastic, but simply uses information about the "goodness" of the function. It is interesting to note that one of our methods has exponential growth depending on the dimension d, as well as ones from [Zhigljavsky, Zilinskas, 2007].

We propose an algorithm for finding the global minimum on a cube for the function (2). It requires $\log(1/\varepsilon)$ iterations and at each iteration the zeroth-order oracle is called $O\left((Ld/\mu)^d\right)$ times. The main idea of this algorithm is that we split a large cube into many small cubes and calculate the function value in each of them. Then we find the minimum value among all the cubes. It can be shown that the real minimum of the problem lies not far from the found point. Therefore, the edge of the original cube can be cut in half, and we can consider a new cube with the center at the found point. Such an algorithm is specifically capable in practice in low-dimensional problems, where the ratio L/μ can be quite large. See Section 1 for details.

For the class of functions (3), we propose a less demanding algorithm, it also has a linear convergence rate $\log(1/\varepsilon)$, but the complexity of its iteration is only $O(d)$. In this case, at each subiteration of the algorithm, we take one of the variables x_i, and equate the rest (x_j with $j \neq i$) with the average value and fix it. And for the variable x_i, we request the value of the function at n points, uniformly distributed from l_i to u_i. Next, we find the minimum among these n points. Then the ith edge of the cube can be halved, and we can consider a new edge centered at the found minimum point. This algorithm shows itself well in practice and for large-scale problems. For more details see Section 2.

The second part follows current trends and works with the concept of gradient reconstruction through finite differences, which is well studied in the literature [Shamir, 2017; Nesterov, Spokoiny, 2017].

As mentioned above, the idea of an inexact oracle is also used here. It is important to note that there are two different random variables ξ^+ and ξ^- in (6). Approximations of this type are referred to as one-point feedback [Akhavan, Pontil, Tsybakov, 2020; Gasnikov et al., 2017; Zhang et al., 2020; Novitskii, Gasnikov, 2021] (compare to two-point feedback from [Beznosikov, Sadiev, Gasnikov, 2020; Beznosikov, Gorbunov, Gasnikov, 2019], where $\xi^+ = \xi^-$). The concept of a one-point feedback is less friendly from the point of view of theoretical analysis, but more realizable from the point of view of practice, since in a real problem it is difficult to achieve the value of a function in two different points with the same realization of a random variable.

In the analysis that is present in the literature (for one or two-point feedbacks) [Risteski, Li, 2016; Akhavan, Pontil, Tsybakov, 2020; Beznosikov, Gorbunov, Gasnikov, 2019; Gasnikov et al., 2017], it is assumed that the noise (or its second moment) is uniformly bounded. In our setting, the noise depends on the distance to the solution — this is the main novelty of our problem statement.

We show that under certain conditions on the noise level and the correct choice of the parameters τ in (6) and the step of the Gradient Descent, it is also possible to achieve a linear convergence rate.

1. “Good” functions

In this section we concentrate on functions from (2). For a better understanding of the method, we present a sketch of the analysis of the algorithm for the one-dimensional case.

Method intuition on a segment

Algorithm 1. BBS

Input: Accuracy ε, parameters L, μ from (2) and bounds l, u.
Let $b := l$, $B := u$ and $n := 2\left\lceil \sqrt{\frac{L}{\mu}} \right\rceil$.
while $B - b \geqslant 2\varepsilon$ **do**

$$i^* := \operatorname*{argmin}_{i \in \{0, \ldots, n\}} f\left(b + i \cdot \frac{(B-b)}{n}\right),$$

$$b := \max\left(b;\, b + \left(i^* - \frac{n}{4}\right) \cdot \frac{(B-b)}{n}\right),$$

$$B := \min\left(B;\, b + \left(i^* + \frac{n}{4}\right) \cdot \frac{(B-b)}{n}\right).$$

end while
Output: $\frac{(B-b)}{2}$.

Suppose we have a function $f\colon [l; u] \to \mathbb{R}$ and it satisfies condition (2). Then let us consider the following procedure, which we call `Bad Binary Search` (or `BBS` for short). The essence of this procedure is very simple. At each iteration of the algorithm, we divide the current segment into n parts and calculate the value of the function at the ends of these segments ($n + 1$ calculations in total). Next, we find the minimum among these $n + 1$ values. It seems that the found point should lie somewhere close to the solution, but this is not entirely true — it depends on n. We claim that it is possible to choose such n that the real minimum will lie in the vicinity of the found minimum. For this we turn to Figure 2: the blue line corresponds to the values of the function, the orange and green lines are the bounding parabolas, the algorithm calculates the value of the function at black points, the minimum level is reached at the point $x^{\min}$, x^* is the real minimum of the function, and the point x^{cl} is the closest point to it. We want $\frac{|x^{\min}-x^*|}{u-l} \leqslant \frac{1}{4}$ (where l, u are current bounds of the segment), which means that we can essentially cut our segment in half and consider a new segment centered at the found minimum.

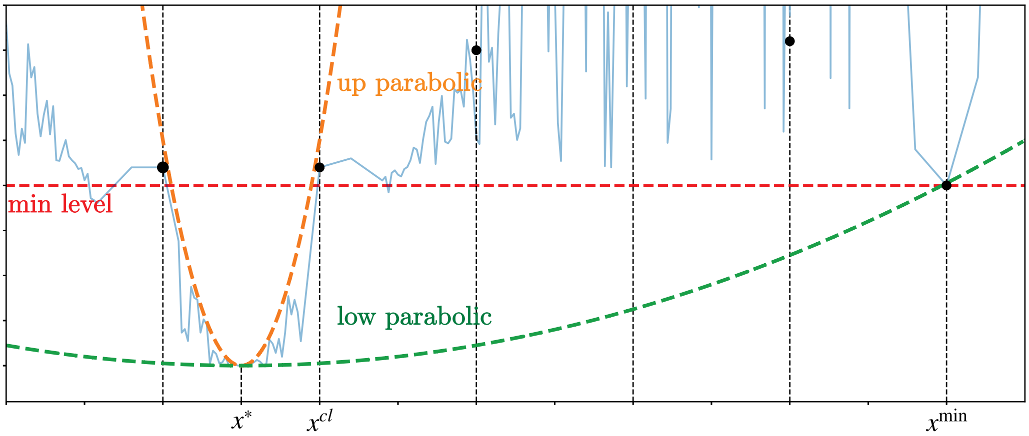

Figure 2. An illustration of the reasoning about the correctness of Algorithm 1. Read the description of notations in the text

Since the value at the point $x^{\min}$ is the minimum among all the others,

$$\frac{\mu}{2}(x^{\min} - x^*)^2 \leqslant f(x^{\min}) \leqslant f(x^{cl}) \leqslant \frac{L}{2}(x^{cl} - x^*)^2.$$

Next, we use the fact that the length of the small segment is $\frac{u-l}{n}$, and the distance between x^* and x_{cl} is no more than $\frac{u-l}{2n}$:

$$\frac{\mu}{2}(x^{\min} - x^*)^2 \leqslant \frac{L}{2}(x^{cl} - x^*)^2 \leqslant \frac{L}{8n^2}(u - l)^2.$$

Hence, we instantly get

$$\frac{|x^{\min} - x^*|}{u - l} \leqslant \frac{1}{2n}\sqrt{\frac{L}{\mu}} \leqslant \frac{1}{4}.$$

The last inequality follows from the requirement $\frac{|x^{\min}-x^*|}{u-l} \leqslant \frac{1}{4}$. Then we get a lower bound on n:

$$n \geqslant 2\left\lceil\sqrt{\frac{L}{\mu}}\right\rceil.$$

Remark. Note that in Algorithm 1 it is necessary to divide n by 4, but n may not be divisible by 4. This does not violate the convergence of the method, but if we take n to be a multiple of 4, then it turns out that at the next iteration of the algorithm we already know the value of the function at half the points, since they coincided with the points from the previous iteration.

Theoretical convergence in $\mathbb{R}^d$

In this part of the work, we present and analyze the algorithm in the case when we work in a space of dimension d. The multidimensional analog of Algorithm 1 in the following way (see Algorithm 2).

Algorithm 2. `Multi BBS`

Input: Accuracy ε, parameters L, μ from (2) and bounds $\vec{l} = (l_1, \ldots, l_d)$, $\vec{u} = (u_1, \ldots, u_d)$.
Let $n := \alpha \left\lceil \sqrt{\frac{dL}{\mu}} \right\rceil$, $b := l$ and $B := u$.
while $\|B - b\| \geqslant \varepsilon$ **do**

$$r := \max_{i \in \{1, \ldots, d\}} \frac{(B_i - b_i)}{n},$$

$$S_j := \left\{0,\ 1,\ \ldots,\ \left\lfloor \frac{(B_j - b_j)}{r} \right\rfloor \right\}, \quad S := S_1 \times \ldots \times S_d,$$

$$i^* := \operatorname*{argmin}_{i \in S} f(b + i \cdot r),$$

$$b := \left\{ b_j := \max\left(b_j;\ b_j + \left(i_j^* - \frac{n}{2\alpha} \right) \cdot r \right),\ j \in \{1, \ldots, d\} \right\},$$

$$B := \left\{ B_j := \min\left(B_j;\ B_j + \left(i_j^* + \frac{n}{2\alpha} \right) \cdot r \right),\ j \in \{1, \ldots, d\} \right\}.$$

end while
Output: $\frac{B-b}{2}$.

Algorithm 2 is more complicated than Algorithm 1, which is due to the fact that we are working with a cube, moreover, this cube may have edges of unequal length. Also, here we generalize the approach for picking n and introduce the parameter $\alpha > 1$.

The idea of `Multi BBS` repeats the idea of the one-dimensional `BBS` algorithm. We also split our cube into small pieces, calculate values at the points on these pieces, and move on to a new smaller cube centered at the minimum of the selected values. It is important to note the size of the pieces into which the original cube is split: all the edges of each small cube have the same length, and this length is determined by the length of the longest edge of the original cube (see the first line of the main loop). This approach allows us to optimize variables that have a large spread (long edge) first. Thereby, if one of the edges of the original cube were much larger than the other ones, then, in fact, only the variable responsible for this edge would be the one to be optimized. The described strategy equalizes the sizes of all cube edges fast, and this is better than splitting large cube edges into small pieces at once.

Theorem 1. `Multi BBS` *algorithm with* $\alpha > 1$ *converges to the global minimum of the function* (2)*. Moreover, the value equal to the maximum length of the cube edge decreases by at least* α *times at each iteration.*

Proof.

Let us introduce the same notation as in the previous subsection: $x_{\min}$ is the point with the minimum value among the selected, x^* is the real minimum of the function, and the point x_{cl} is the closest point to it. Then

$$\frac{\mu}{2} \sum_{i=1}^{d} (x_i^{\min} - x_i^*)^2 \leqslant f(x^{\min}) \leqslant f(x^{cl}) \leqslant \frac{L}{2} \sum_{i=1}^{d} (x_i^{cl} - x_i^*)^2.$$

Since the edge of a small cube is at most r, then for all i

$$(x_i^{cl} - x_i^*)^2 \leqslant \frac{r^2}{4},$$

and

$$\frac{\mu}{2}\sum_{i=1}^{d}(x_i^{\min} - x_i^*)^2 \leqslant \frac{dLr^2}{8} = \frac{dL}{8n^2}\left(\max_{i\in\{1,\ldots,d\}}[B_i - b_i]\right)^2.$$

It is easy to see that, for $n = \alpha\left\lceil\sqrt{\frac{dL}{\mu}}\right\rceil$, we get

$$\sum_{i=1}^{d}(x_i^{\min} - x_i^*)^2 \leqslant \frac{1}{4\alpha^2}\left(\max_{i\in\{1,\ldots,d\}}[B_i - b_i]\right)^2,$$

whence it follows that for all i

$$\frac{|x_i^{\min} - x_i^*|}{\max\limits_{i\in\{1,\ldots,d\}}[B_i - b_i]} \leqslant \frac{1}{2\alpha}.$$

This inequality ensures that the maximum edge length of the new cube is (at least) α times less than the maximum edge length of the old one. □

The theorem implies the following corollary on the complexity of the algorithm:

Corollary 1. *Algorithm* 2 *requires* $O(\log_\alpha(1/\varepsilon))$ *iterations to find a solution* (*in terms of Algorithm* 2). *Moreover, the oracle complexity of each iteration is* $O\left(\left(\alpha\sqrt{\frac{dL}{\mu}}\right)^d\right)$.

Proof.

To prove the first statement, we write the simple chain for B and b after T iteration:

$$\|B^T - b^T\|^2 \leqslant d\left(\max_{i\in\{1,\ldots,d\}} B_i^T - b_i^T\right)^2 \leqslant d\left(\frac{\max\limits_{i\in\{1,\ldots,d\}}[u_i - l_i]}{\alpha^T}\right)^2 \leqslant \varepsilon^2,$$

where u and l are starting cube boundaries. This implies the required statement.

The second statement follows from the fact that in the worst case (when all the edges of the cube are equal) we need to calculate the value of the function at $O(n^d)$ points. □

The complexity of one iteration increases dramatically with the growth of the dimension; therefore, this method is proposed to be used for solving low-dimensional problems.

Small dimension numerical experiments

We give examples of how the algorithm works on low-dimensional problems: one-dimensional and two-dimensional. First, consider the following function:

$$f(x) = 10(x-2)^2 - 4\cos[17(x-2)] + 4, \quad x \in [0, 6{,}5]. \tag{7}$$

The global minimum on this segment is the point $x^* = 2$. We take $L = 600$, $\mu = 10$. The starting point is the center of the segment $x^0 = 3{,}25$. `Multi BBS` algorithm starts with $\alpha = 1{,}5, 2, 3, 4$. The convergence of the algorithm is shown in Figure 3, *a*.

Next, we work with a 2-dimensional problem — the Levy function:

$$f(x, y) = \sin^2[3\pi(x-2{,}7)] + (x-3{,}7)^2(1+\sin^2[3\pi(y-0{,}3)]) + (y-1{,}3)^2(1+\sin^2[2\pi(y-0{,}3)]), \tag{8}$$

where $x, y \in [-10, 10]$. The optimal point: $f(3{,}7,\ 1{,}3) = 0$ and $L = 150$, $\mu = 1$. For the trajectory of convergence, see Figure 3, *b*.

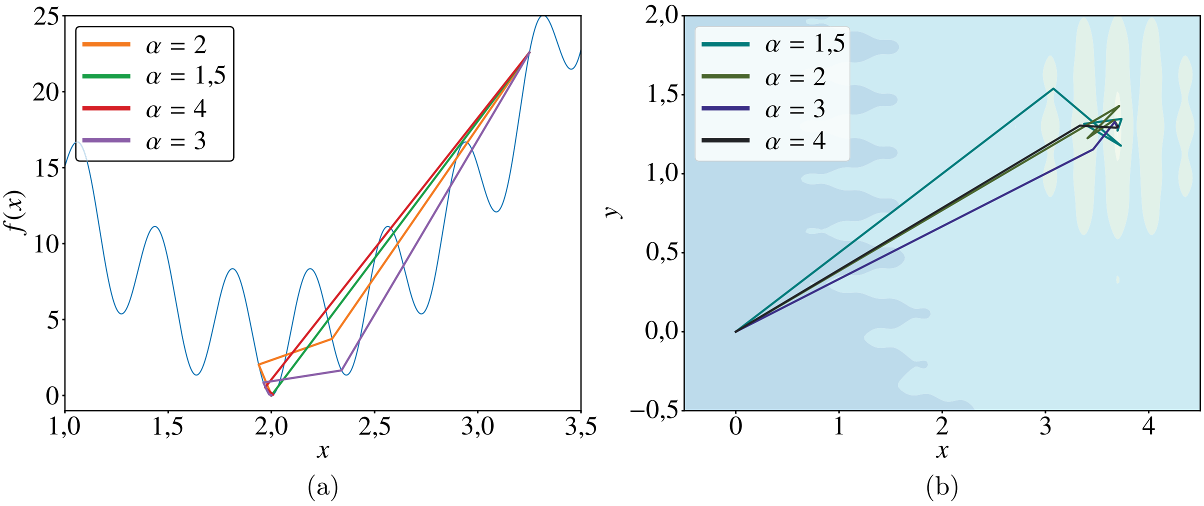

Figure 3. Convergence of the `Multi BBS` algorithm for problems of dimensions 1 and 2: a) problem (7); b) problem (8)

2. Very “good” functions

In this section we analyze functions $f(x)$ which meet the requirements (3). For such a problem statement, we present a less demanding version of the `BBS` algorithm.

Algorithm 3. `Direction BBS`

Input: Accuracy ε, parameters M, Δ from (3) and bounds $l = (l_1, \ldots, l_d)$, $u = (u_1, \ldots, u_d)$.
Let $n := 15$, $b := l$, $B := u$, and $m := \frac{l+u}{2}$.
while $\|B - u\| \geqslant 2\varepsilon$ **do**
  for $i = 1, \ldots, d$ **do**

$$R := \max_{i \in \{1, \ldots, d\}} B_i - b_i,$$

$$j^* := \operatorname*{argmin}_{j=0, \ldots, n} f\left(m_1, \ldots, m_{i-1}, b_i + j \cdot \frac{(B_i - b_i)}{n}, m_{i+1}, \ldots, m_d\right),$$

$$m_i := b_i + j^* \cdot \frac{(B_i - b_i)}{n},$$

$$b_i := \max\left(b_i; m_i - \frac{R}{3}\right),$$

$$B_i := \min\left(B_i; m_i + \frac{R}{3}\right).$$

  end for
end while
Output: $\frac{B-b}{2}$.

This algorithm no longer draws a “grid” over the entire cube. In this case, at each external iteration (while), we go through all the variables in turn. At the inner iteration (for), we consider only one variable, while the rest are fixed equal to m_j. Then we do the procedure in a similar way to `BBS` — we divide the current edge and do a one-dimensional search for the minimum among the calculated points. Due to the fact that the problem is very “good”, this approach allows the method to converge. This is what the next section is about.

Theoretical analysis

Theorem 2. `Direction BBS` *algorithm converges to the global minimum of the function* (3). *Moreover, the value equal to the maximum length of the cube edge decreases by at least* $\frac{3}{2}$ *times at each main iteration* (*an outer loop with while*).

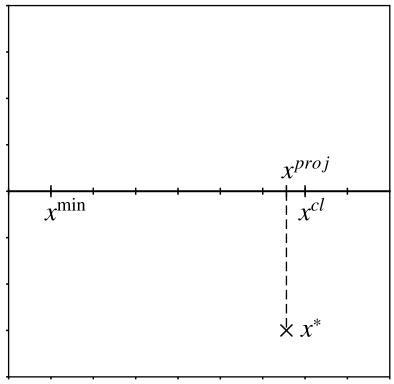

Figure 4. Illustration for the proof of Theorem 2

Proof.

Consider the ith iteration of inner loop (with for). Let $x^{\min} = \left(m_1, \ldots, m_{i-1}, b_i + j^* \cdot \frac{(B_i - b_i)}{n}, m_{i+1}, \ldots, m_d\right)$, x^* be the real minimum, $x^{proj} = \left(m_1, \ldots, m_{i-1}, x_i^*, m_{i+1}, \ldots, m_d\right)$, and x^{cl} be the closest point to x^{proj}, where we calculate the value of the function (see Figure 4). Then the following inequality holds:

$$f(x^{\min}) \leqslant f(x^{cl}).$$

Using (3), we can note that

$$\left(\frac{M}{2} + \delta(x^{\min})\right)\|x^{\min} - x^*\|_2^2 \leqslant \left(\frac{M}{2} + \delta(x^{cl})\right)\|x^{cl} - x^*\|_2^2.$$

By a definition of Euclidean norm,

$$\left(\frac{M}{2} + \delta(x^{\min})\right)\left((x_i^{\min} - x_i^{proj})^2 + \|x^{proj} - x^*\|^2\right) \leqslant \left(\frac{M}{2} + \delta(x^{cl})\right)\left((x_i^{cl} - x_i^{proj})^2 + \|x^{proj} - x^*\|^2\right).$$

As we know, $|\delta(x)|$ is not greater than Δ. Therefore,

$$\left(\frac{M}{2} - \Delta\right)\left((x_i^{\min} - x_i^{proj})^2 + \|x^{proj} - x^*\|^2\right) \leqslant \left(\frac{M}{2} + \Delta\right)\left((x_i^{cl} - x_i^{proj})^2 + \|x^{proj} - x^*\|^2\right). \tag{9}$$

Since x_i^{cl} is the closest to x_i^{proj}, it follows that

$$(x_i^{cl} - x_i^{proj})^2 \leqslant \left(\frac{B_i - r_i}{2n}\right)^2 \leqslant \left(\frac{R}{2n}\right)^2. \tag{10}$$

In the last inequality we use the definition of R from Algorithm 3. By (10) and some simple transformation of (9), we get

$$\left(\frac{M}{2} - \Delta\right)(x_i^{\min} - x_i^{proj})^2 \leqslant \left(\frac{M}{2} + \Delta\right)\left(\frac{R}{2n}\right)^2 + 2\Delta\|x^{proj} - x^*\|^2. \tag{11}$$

Also, it is quite obvious that

$$\|x^{proj} - x^*\|^2 \leqslant \sum_{j\in\{1,\ldots,d\}\setminus\{i\}} \left(\frac{B_j - b_j}{2}\right)^2 \leqslant \frac{(d-1)R^2}{4}. \tag{12}$$

Combining inequalities (11) and (12),

$$\left(\frac{M}{2} - \Delta\right)(x_i^{\min} - x_i^{proj})^2 \leqslant \left(\frac{M}{2} + \Delta\right)\left(\frac{R}{2n}\right)^2 + \Delta\frac{(d-1)R^2}{2}.$$

Then we get the following expression:

$$(x_i^{\min} - x_i^{proj})^2 \leqslant \frac{(M + 2\Delta)}{(M - 2\Delta)}\left(\frac{R}{2n}\right)^2 + \Delta\frac{(d-1)R^2}{M - 2\Delta}.$$

Substituting boundaries for Δ from (3) and $n = 15$, we have

$$(x_i^{\min} - x_i^{proj})^2 \leqslant \frac{1 + \frac{1}{8(d-1)}}{1 - \frac{1}{8(d-1)}}\left(\frac{R}{30}\right)^2 + \frac{R^2}{16 - \frac{2}{d-1}}.$$

Note that $d - 1 \geqslant 1$, then

$$(x_i^{\min} - x_i^{proj})^2 \leqslant \left(\frac{1}{700} + \frac{1}{14}\right)R^2 \leqslant \frac{51R^2}{700}. \tag{13}$$

Summing up (10) and (13), we get

$$|x_i^{\min} - x_i^{cl}| \leqslant |x_i^{\min} - x_i^{proj}| + |x_i^{cl} - x_i^{proj}| \leqslant \left(\frac{1}{30} + \sqrt{\frac{51}{700}}\right)R \leqslant \frac{R}{3}.$$

This means that

$$\frac{|x_i^{\min} - x_i^{cl}|}{\max\limits_{j\in\{1,\ldots,d\}} B_j - b_j} \leqslant \frac{1}{3}.$$

This inequality ensures that the maximum edge length of the new cube is (at least) $\frac{3}{2}$ times less than the maximum edge length of the old one. □

Corollary 2. *Algorithm* 3 *requires* $O\left(\log_{3/2}(1/\varepsilon)\right)$ *outer iterations to find a solution. Moreover, the oracle complexity of each iteration is* $O(d)$.

Proof.

To prove the first statement, we write the simple chain for B and b after T iteration:

$$\|B^T - b^T\|^2 \leqslant d\left(\max_{i\in\{1,\ldots,d\}} B_i^T - b_i^T\right)^2 \leqslant d\left(\frac{\max\limits_{i\in\{1,\ldots,d\}}[u_i - l_i]}{\left(\frac{3}{2}\right)^T}\right)^2 \leqslant \varepsilon^2,$$

where u and l are starting cube boundaries. This implies the required statement.

The second statement follows immediately from the description of the algorithm. □

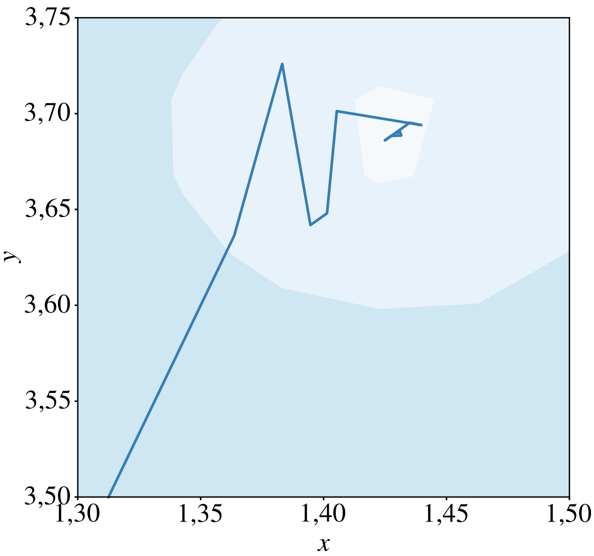

Figure 5. Convergence trajectory of `Direction BBS` for $d = 2$

One can note that if the current ith variable has a small edge compared to the rest, then we essentially do the internal iteration over it in vain, because we shrink the current cube edge using the length of the maximum one. Therefore, the `Direction BBS` algorithm can be modified as follows: to remove the inner loop (with for) and to iterate the main loop (with while) in the direction that has the longest edge at the moment, this approach eliminates the case where we wastefully consider a small edge in the presence of a large one.

Practical application

The function $f(x)$ from (3) is created in the following way: we select some point x^*, and also generate the values of the function $\delta(x)$ uniformly on the segment $[-\Delta, \Delta]$ and independent of values at other points, then it is easy to construct $f(x)$ with $M = 20$. In the experiment, we do not restore the $\delta(x)$ function completely, we generate values only at the required points.

We start with a problem of dimension 2. The optimal point $x^* = (1{,}43,\ 3{,}69)$, bounds $l_i = -10$, $u_i = 10$ (for $i = 1,\ 2$). The convergence of the method is shown in Figure 5.

Next, we used our algorithm for 10- and 100-dimensional problems with $x^* = (1, \ldots, 1)$, $l_i =$ $= -10$, $u_i = 10$ (for all i). The result of the convergence of the display is shown in Figure 6.

The results confirm the linear convergence stated in the theory.

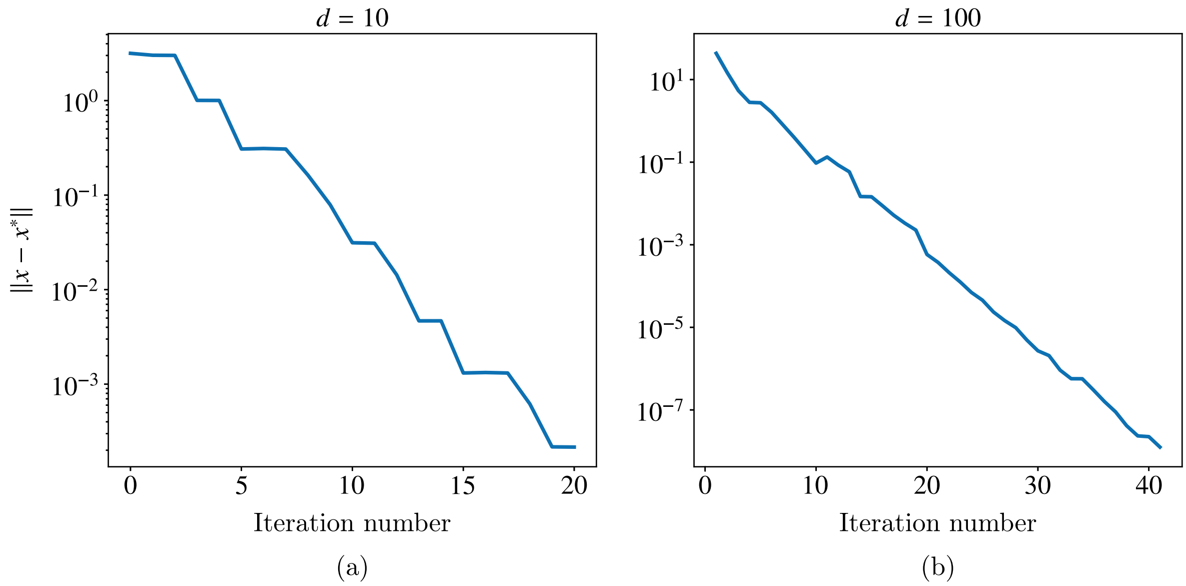

Figure 6. Convergence of the `Direction BBS` method in the case where the dimension of the problem is: a) 10; b) 100

Now let us move on to the second part of the paper.

3. Another view of “good” functions

Recall that in this section we minimize the function $\frac{1}{2}(x-x^*)^T A(x-x^*)$ with a positive definite matrix A. One can note that our problem is μ-strongly-convex and has L Lipshitz gradient. We have access to noise oracle:

$$f(x, \xi) = \frac{1}{2}(x-x^*)^T A(x-x^*) + (\xi + \delta(x))\|x-x^*\|,$$

where the random variable ξ does not depend on the point x and is generated randomly so that $\mathbb{E}\xi = 0$ and $\mathbb{E}\xi^2 \leqslant \sigma^2$, $|\delta(x)| \leqslant \Delta$ for all x. Then, with the help of such an oracle, one can restore the gradient using (6) in the following form:

$$g(x, \xi^\pm, \tau, e) = n\langle A(x-x^*), e\rangle e + \frac{n}{2\tau}\left((\xi^+ + \delta(x+\tau e))\|x+\tau e - x^*\| - \right.$$
$$\left. - (\xi^- + \delta(x-\tau e))\|x-\tau e - x^*\|\right) e, \quad (14)$$

where e is a random vector uniformly distributed over the Euclidean sphere. Then with this oracle we can run the classic Gradient Descent.

Algorithm 4. `zoGD`

Input: Number of iterations K, parameters γ_k, τ_k.
for $k = 1, \ldots, K$ **do**
  Generate independently $\xi_k^\pm$, e_k,
  $x_{k+1} = x_k - \gamma_k g(x_k, \xi_k^\pm, \tau_k, e_k)$.
end for
Output: x_{K+1}.

Theoretical analysis

Here we prove the convergence theorem and corollaries.

Theorem 3. *The following estimate for the iteration of Algorithm 4 is valid*

$$\mathbb{E}\left[\|x_{k+1}-x^*\|^2\right] \leqslant \left(1 - \gamma_k\mu + \frac{5d^2\gamma_k^2(\Delta^2+\sigma^2)}{\tau_k^2}\right)\mathbb{E}\left[\|x_k - x^*\|^2\right] +$$
$$+ \frac{2d\gamma_k\Delta}{\tau_k}\mathbb{E}\left[\|x_k - x^*\|\right] + 2d\gamma_k\Delta + 5d^2\gamma_k^2(\Delta^2+\sigma^2).$$

Proof.

We start with the step of our method:

$$\|x_{k+1}-x^*\|^2 = \|x_k - \gamma_k g(x_k, \xi_k^\pm, \tau_k, e_k) - x^*\|^2 =$$
$$= \|x_k - x^*\|^2 - 2\gamma_k\langle g(x_k, \xi_k^\pm, \tau_k, e_k), x_k - x^*\rangle + \gamma_k^2\|g(x_k, \xi_k^\pm, \tau_k, e_k)\|^2.$$

Taking the full expectation and taking into account that $z_k - z^*$ does not depend on e_k, ξ_k, we get

$$\mathbb{E}\left[\|x_{k+1}-x^*\|^2\right] = \mathbb{E}\left[\|x_k-x^*\|^2\right] - 2\gamma_k\mathbb{E}\left[\langle g(x_k, \xi_k^\pm, \tau_k, e_k), x_k - x^*\rangle\right] + \gamma_k^2\mathbb{E}\left[\|g(x_k, \xi_k^\pm, \tau_k, e_k)\|^2\right] =$$
$$= \mathbb{E}\left[\|x_k-x^*\|^2\right] - 2\gamma_k\mathbb{E}\left[\langle\mathbb{E}_{\xi_k, e_k}\left[g(x_k, \xi_k^\pm, \tau_k, e_k)\right], x_k - x^*\rangle\right] + \gamma_k^2\mathbb{E}\left[\|g(x_k, \xi_k^\pm, \tau_k, e_k)\|^2\right]. \quad (15)$$

We use that $\mathbb{E}[d\langle s, e\rangle e] = s$. Next, we need to estimate $\mathbb{E}\left[\langle g(x_k, \xi_k^{\pm}, \tau_k, e_k), x_k - x^*\rangle\right]$ and $\mathbb{E}\left[\|g(x_k, \xi_k^{\pm}, \tau_k, e_k)\|^2\right]$. We start with $\mathbb{E}\left[\langle g(x_k, \xi_k^{\pm}, \tau_k, e_k), x_k - x^*\rangle\right]$:

$$\begin{aligned}
&\mathbb{E}_{\xi_k, e_k}\left[g(x_k, \xi_k^{\pm}, \tau_k, e_k)\right] = \mathbb{E}_{e_k}\left[d\langle A(x_k - x^*), e_k\rangle e_k\right] + \\
&\quad + \frac{d}{2\tau_k}\mathbb{E}_{\xi_k, e_k}\left[\left((\xi_k^+ + \delta(x_k + \tau_k e_k))\|x_k + \tau_k e_k - x^*\| - (\xi_k^- + \delta(x_k - \tau_k e_k))\|x_k - \tau_k e_k - x^*\|\right) e_k\right] = \\
&\quad = A(x_k - x^*) + \frac{d}{2\tau_k}\mathbb{E}_{e_k}\left[\left(\delta(x_k + \tau_k e_k)\|x_k + \tau_k e_k - x^*\| - \delta(x_k - \tau_k e_k)\|x_k - \tau_k e_k - x^*\|\right) e_k\right]. \quad (16)
\end{aligned}$$

Let's work with $\mathbb{E}\left[\|g(x_k, \xi_k^{\pm}, \tau_k, e_k)\|^2\right]$:

$$\begin{aligned}
&\mathbb{E}\left[\|g(x_k, \xi_k^{\pm}, \tau_k, e_k)\|^2\right] = \mathbb{E}\Big[\mathbb{E}_{e_k}\left[\|d\langle A(x_k - x^*), e_k\rangle e_k\|^2\right] + \\
&\quad + \frac{d}{2\tau_k}\left((\xi_k^+ + \delta(x_k + \tau_k e_k))\|x_k + \tau_k e_k - x^*\| - (\xi_k^- + \delta(x_k - \tau_k e_k))\|x_k - \tau_k e_k - x^*\|\right) e_k\|^2\Big] \leqslant \\
&\quad \leqslant 5n^2\mathbb{E}\left[\mathbb{E}_{e_k}\left[\|d\langle A(x_k - x^*), e_k\rangle e_k\|^2\right]\right] + \frac{5d^2}{4\tau_k^2}\mathbb{E}\left[(\xi_k^+)^2\|x_k + \tau e_k - x^*\|^2\|e_k\|^2\right] + \\
&\quad + \frac{5d^2}{4\tau_k^2}\mathbb{E}\left[\delta_k^2(x_k + \tau_k e_k)\|x_k + \tau e_k - x^*\|^2\|e_k\|^2\right] + \frac{5d^2}{4\tau_k^2}\mathbb{E}\left[(\xi_k^-)^2\|x - \tau_k e_k - x^*\|^2\|e_k\|^2\right] + \\
&\quad + \frac{5d^2}{4\tau_k^2}\mathbb{E}\left[\delta^2(x_k - \tau_k e_k)\|x_k - \tau_k e_k - x^*\|^2\|e_k\|^2\right].
\end{aligned}$$

Next, we use Lemma B.10 from [Bogolubsky et al., 2016]: $\mathbb{E}[|\langle s, e\rangle|^2] = \frac{1}{d}\|s\|^2$ for some vector s and independent e — random vector uniformly distributed on the Euclidean sphere.

$$\begin{aligned}
&\mathbb{E}\left[\|g(x_k, \xi_k^{\pm}, \tau_k, e_k)\|^2\right] \leqslant 5d\mathbb{E}\left[\|A(x_k - x^*)\|^2\right] + \frac{5d^2}{4\tau_k^2}\mathbb{E}\left[\mathbb{E}_{\xi_k}\left[(\xi_k^+)^2\right]\|x_k + \tau_k e_k - x^*\|^2\right] + \\
&+ \frac{5d^2\Delta^2}{4\tau^2}\mathbb{E}\left[\|x_k + \tau_k e_k - x^*\|^2\right] + \frac{5d^2}{4\tau_k^2}\mathbb{E}\left[\mathbb{E}_{\xi_k}\left[(\xi_k^-)^2\right]\|x_k - \tau_k e_k - x^*\|^2\right] + \frac{5d^2\Delta^2}{4\tau_k^2}\mathbb{E}\left[\|x_k - \tau_k e_k - x^*\|^2\right] \leqslant \\
&\quad \leqslant 5n\mathbb{E}\left[\|A(x_k - x^*)\|^2\right] + \frac{5d^2\sigma^2}{2\tau_k^2}\mathbb{E}\left[\|x_k - x^*\|^2 + \|\tau_k e_k\|^2\right] + \frac{5d^2\Delta^2}{2\tau_k^2}\mathbb{E}\left[\|x_k - x^*\|^2 + \|\tau_k e_k\|^2\right] + \\
&\quad + \frac{5d^2\sigma^2}{2\tau_k^2}\mathbb{E}\left[\|x_k - x^*\|^2 + \|\tau_k e_k\|^2\right] + \frac{5d^2\Delta^2}{2\tau_k^2}\mathbb{E}\left[\|x_k - x^*\|^2 + \|\tau_k e_k\|^2\right] \leqslant \\
&\quad \leqslant 5d\mathbb{E}\left[\|A(x_k - x^*)\|^2\right] + \frac{5d^2(\Delta^2 + \sigma^2)}{\tau_k^2}\mathbb{E}\left[\|x_k - x^*\|^2\right] + 5d^2(\Delta^2 + \sigma^2). \quad (17)
\end{aligned}$$

Then we combine (15), (16), and (17):

$$\begin{aligned}
&\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] \leqslant \mathbb{E}\left[\|x_k - x^*\|^2\right] - 2\gamma_k\mathbb{E}\left[\langle A(x_k - x^*), x_k - x^*\rangle\right] - \\
&\quad - \frac{d\gamma_k}{\tau_k}\mathbb{E}\left[\langle\mathbb{E}_{e_k}\left[\left(\delta(x_k + \tau_k e_k)\|x_k + \tau_k e_k - x^*\| - \delta(x_k - \tau_k e_k)\|x_k - \tau_k e_k - x^*\|\right) e_k\right], x_k - x^*\rangle\right] + \\
&\quad + 5d\gamma_k^2\mathbb{E}\left[\|A(x_k - x^*)\|^2\right] + \frac{5d^2\gamma_k^2(\Delta^2 + \sigma^2)}{\tau_k^2}\mathbb{E}\left[\|x_k - x^*\|^2\right] + 5d^2\gamma_k^2(\Delta^2 + \sigma^2).
\end{aligned}$$

With $\nabla f(x_k) = A(x_k - x^*)$, we have

$$\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] \leqslant \left(1 + \frac{5d^2\gamma_k^2(\Delta^2 + \sigma^2)}{\tau_k^2}\right)\mathbb{E}\left[\|x_k - x^*\|^2\right] -$$
$$- 2\gamma_k\mathbb{E}\left[\langle\nabla f(x_k),\, x_k - x^*\rangle\right] + 5d\gamma_k^2\mathbb{E}\left[\|\nabla f(x_k)\|^2\right] + \frac{n\gamma_k}{\tau_k}\mathbb{E}\left[\left\|\mathbb{E}_{e_k}\left[\left(\delta(x_k + \tau_k e_k)\|x_k + \tau_k e_k - x^*\| -\right.\right.\right.\right.$$
$$\left.\left.\left.\left.- \delta(x_k - \tau_k e_k)\|x_k - \tau_k e_k - x^*\|\right) e_k\right]\right\| \cdot \|x_k - x^*\|\right] + 5d^2\gamma_k^2(\Delta^2 + \sigma^2). \quad (18)$$

We work with $\Upsilon_k = \left\|\mathbb{E}_{e_k}\left[\left(\delta(x_k + \tau_k e_k)\|x_k + \tau_k e_k - x^*\| - \delta(x_k - \tau_k e_k)\|x_k - \tau_k e_k - x^*\|\right) e_k\right]\right\|$:

$$\Upsilon_k \leqslant \mathbb{E}_{e_k}\left[|\delta(x_k + \tau_k e_k)| \cdot \|x_k + \tau_k e_k - x^*\| + |\delta(x_k - \tau_k e_k)| \cdot \|x_k - \tau_k e_k - x^*\|\right] \leqslant$$
$$\leqslant \Delta\mathbb{E}_{e_k}\left[\|x_k + \tau_k e_k - x^*\| + \|x_k - \tau_k e_k - x^*\|\right] \leqslant 2\Delta\mathbb{E}_{e_k}\left[\|x_k - x^*\| + \tau_k\right]. \quad (19)$$

Connecting with (18), we get

$$\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] \leqslant \left(1 + \frac{5d^2\gamma_k^2(\Delta^2 + \sigma^2)}{\tau_k^2}\right)\mathbb{E}\left[\|x_k - x^*\|^2\right] -$$
$$- 2\gamma_k\mathbb{E}\left[\langle\nabla f(x_k),\, x_k - x^*\rangle\right] + 5d\gamma_k^2\mathbb{E}\left[\|\nabla f(x_k)\|^2\right] + \frac{2d\gamma_k\Delta}{\tau_k}\mathbb{E}\left[\|x_k - x^*\|\right] + 2d\gamma_k\Delta + 5d^2\gamma_k^2(\Delta^2 + \sigma^2).$$

Using μ-strong convexity and L-smoothness of f, we have

$$\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] \leqslant \left(1 - \gamma_k\mu + \frac{5d^2\gamma_k^2(\Delta^2 + \sigma^2)}{\tau_k^2}\right)\mathbb{E}\left[\|x_k - x^*\|^2\right] -$$
$$- 2\gamma_k(1 - 5d\gamma_k L)\mathbb{E}\left[f(x_k) - f(x^*)\right] + \frac{2d\gamma_k\Delta}{\tau_k}\mathbb{E}\left[\|x_k - x^*\|\right] + 2d\gamma_k\Delta + 5d^2\gamma_k^2(\Delta^2 + \sigma^2).$$

With $\gamma_k \leqslant \frac{1}{5dL}$

$$\mathbb{E}\left[\|x_{k+1} - x^*\|^2\right] \leqslant \left(1 - \gamma_k\mu + \frac{5d^2\gamma_k^2(\Delta^2 + \sigma^2)}{\tau_k^2}\right)\mathbb{E}\left[\|x_k - x^*\|^2\right] +$$
$$+ \frac{2d\gamma_k\Delta}{\tau_k}\mathbb{E}\left[\|x_k - x^*\|\right] + 2d\gamma_k\Delta + 5d^2\gamma_k^2(\Delta^2 + \sigma^2).$$

□

Corollary 3. *If* $\Delta = 0$, $\gamma_k = \gamma \leqslant \frac{1}{5dL}$ *and* $\tau_k \geqslant \sqrt{\frac{2d\sigma^2}{\mu L}}$ *then*

$$\mathbb{E}\left[\|x_{K+1} - x^*\|^2\right] \leqslant \left(1 - \frac{\gamma\mu}{2}\right)^K \|x_0 - x^*\|^2 + \frac{10d^2\gamma\sigma^2}{\mu}.$$

Additionally, if

$$\gamma = \min\left\{\frac{1}{5dL};\ \frac{2\ln\left(\max\left\{2;\ \frac{\mu^2\|z^0 - z^*\|^2 K}{20d^2\sigma^2}\right\}\right)}{\mu K}\right\},$$

then

$$\mathbb{E}\left[\|x_{K+1} - x^*\|^2\right] \leqslant \widetilde{O}\left(\exp\left(-\frac{\mu K}{20dL}\right)\|x_0 - x^*\|^2 + \frac{20d^2\sigma^2}{\mu^2 K}\right).$$

It gives the results similar to SGD convergence.

Practical application

We consider oracle (4) with $n = 50$, $L = 100$, $\mu = 1$ and $\sigma = 1, 2, 5, 10, 20, 100$, $\Delta = 0$. In Algorithm 4 we use constant $\gamma = \frac{1}{dL}$ and $\tau = \sqrt{\frac{2d\sigma^2}{\mu L}}$. See the convergence of Algorithm 4 with different σ in Figure 7.

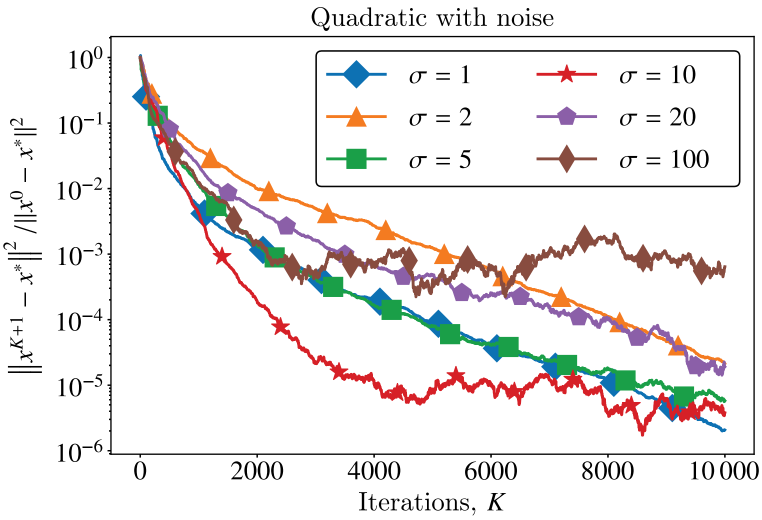

Figure 7. Convergence of Algorithm 4 with oracle (4) at different noise levels

References

Akhavan A., Pontil M., Tsybakov A. B. Exploiting higher order smoothness in derivative-free optimization and continuous bandits // arXiv preprint. — 2020. — https://arxiv.org/pdf/2006.07862.pdf

Beznosikov A., Gorbunov E., Gasnikov A. Derivative-Free Method For Composite Optimization With Applications To Decentralized Distributed Optimization // arXiv preprint. — 2019. — https://arxiv.org/pdf/1911.10645.pdf

Beznosikov A., Sadiev A., Gasnikov A. Gradient-Free Methods for Saddle-Point Problem // arXiv preprint. — 2020. — https://arxiv.org/pdf/2005.05913.pdf

Bogolubsky L., Dvurechensky P., Gasnikov A., Gusev G., Nesterov Yu., Raigorodskii A., Tikhonov A., Zhukovskii M. Learning supervised pagerank with gradient-based and gradient-free optimization methods // arXiv preprint. — 2016. — https://arxiv.org/pdf/1603.00717.pdf

Boyd S. P., Vandenberghe L. Convex optimization. — Cambridge: Cambridge University Press, 2018.

Gasnikov A. V., Krymova E. A., Lagunovskaya A. A., Usmanova I. N., Fedorenko F. A. Stochastic online optimization. Single-point and multi-point non-linear multi-armed bandits. Convex and strongly-convex case // Automation and remote control. — 2017. — Vol. 78, No. 2. — P. 224–234.

Nesterov Yu., Spokoiny V. G. Random Gradient-Free Minimization of Convex Functions // Foundations of Computational Mathematics. — 2017. — Vol. 17, No. 2. — P. 527–566. — https://doi.org/10.1007/s10208-015-9296-2

Nocedal J., Wright S. J. Numerical Optimization. — 2nd edition. — New York, NY, USA: Springer, 2006.

Novitskii V., Gasnikov A. Improved Exploiting Higher Order Smoothness in Derivative-free Optimization and Continuous Bandit // arXiv preprint. — 2021. — https://arxiv.org/pdf/2101.03821.pdf

Polyak B. T. Introduction to optimization // Optimization Software, Inc, New York. — 1987.

Risteski A., Li Y. Algorithms and matching lower bounds for approximately-convex optimization // Advances in Neural Information Processing Systems 29 / D. D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, R. Garnett (eds.). — Curran Associates, Inc., 2016. — https://proceedings.neurips.cc/paper/2016/file/186fb23a33995d91ce3c2212189178c8-Paper.pdf

Shamir O. An Optimal Algorithm for Bandit and Zero-Order Convex Optimization with Two-Point Feedback // Journal of Machine Learning Research. — 2017. — Vol. 18, No. 52. — P. 1–11.

Shor N. Z. Minimization methods for non-differentiable functions. — Springer Science & Business Media, 2012. — Vol. 3.

Singer Y., Vondrak J. Information-theoretic lower bounds for convex optimization with erroneous oracles // Advances in Neural Information Processing Systems 28 / C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, R. Garnett (eds.). — Curran Associates, Inc., 2015. — P. 3204–3212. — http://papers.nips.cc/paper/5841-information-theoretic-lower-bounds-for-convex-optimization-with-erroneous-oracles.pdf

Zhang Y., Zhou Y., Ji K., Zavlanos M. M. Improving the Convergence Rate of One-Point Zeroth-Order Optimization using Residual Feedback // arXiv preprint. — 2020. — https://arxiv.org/pdf/2006.10820.pdf

Zhigljavsky A., Zilinskas A. Stochastic global optimization. — Springer Science & Business Media, 2007. — Vol. 9.