Contents

We consider some new approach to method convex 2-dimensional optimization on a fixed square recently proposed by Yu. E. Nesterov (see ex.1.5 in https://arxiv.org/pdf/1711.00394.pdf). The method can be interested for to solve a dual problem for a convex problem with two functional constraints. The idea of the method consists in the narrowing of the domain until we achieve an acceptable quality of the solution. This method has to search minimum on the separating segments and we propose some non-depended on required accuracy stop condition for it. Estimations for the iterations number are proved for the cases of smooth and non-smooth functions. Besides there is an experimental comparison of our method with other inexact methods of convex optimization such as ellipsoid method with epsilon-subgradient and primal and fast gradient method with delta-L-oracle. The experiments were made on dual problems for problems with two constraints As a result, our method with provided in this work strategy has the best result.

1 Introduction

In this paper we research one method of optimization on a square in $\mathbb{R}^2$. The method was offered by Nesterov (see ex. 4 from [1]). interested for to solve a dual problem for a convex problem with two functional constraints. This method is reffered as Halving Square Method.

The idea of the method consists in the narrowing of the domain until we achieve an acceptable quality of the solution. This method has to search minimum on the separating segments. In the next section there is description of method and pseudocode for it.

In the paper [2] there are some results for this method. Namely, there is an estimate for iterations number. Additionally, the authors provided strategy for subproblem on segment. Also there are comparison of this method with method of ellipsoids. Discussed method showed better results on time.

But the provided strategy has several disadvantages. Firstly, the required accuracy for the problem on segment significantly depends on the required accuracy for initial task on segment. It is essential disadvantage when one want to solve initial task extra accurately. Secondly, the method has convergence only on function and is able to not converge to solution on argument.

In this paper we continue this research and provide a new strategy for separating segments. According to it there is stop condition for one-dimensional problem. This stop condition does not depend on required initial accuracy

and the HSM with this strategy converges to the solution on argument. The discussion of this strategy is in the section 4.

Moreover, the new estimates for iterations number was provided in this work. There are estimates for smooth and non-smooth problem when on each iteration one chooses rectangle that include the global solution. Additionally, there are theoretical and experimental comparisons of this estimates.

Also we carried out some numerical expirements for to compare the Halving Square Method with other methods. Firstly, it is comparison of different strategies for the Halving Square Method on problems when we can calculate object function's value analitacally. Secondly, it is comparison on dual problem when we can not do it. In this case we compare our method with other such inexact methods as primal gradient method, fast gradient method and ellipsoid method.

This experiments showed that the Halving Square Method with provided in this work strategy is optimal method for to solve dual problem with high accuracy. And the gain of using this method grows with the growth of required accuracy.

2 Method Description

Let's consider a following task:

$$\min_{(x,y)} \{f(x,y)|(x,y) \in Q\},$$

where f is a convex function, Q - is a square on the plane.

Let's consider a following method. One solves task of minimization for a function $g(x) = f\left(x, y_0 = \frac{a}{2}\right)$ on a segment $[0, a]$ with an accuracy δ on function. After that one calculates a sub-gradient in a received point and chooses the rectangle which the sub-gradient "does not look" in. Similar actions are repeated for a vertical segment. As a result we have the square decreased twice. Let's find a possible value of error δ_0 for task on segment and a sufficient iteration's number N to solve the initial task with accuracy ϵ on function.

Let's describe an algorithm formally. See pseudo-code 1.

Algorithm 1 Algorithm of the method

function METHOD(convex function f, square $Q = [a, b] \times [c, d]$)
  $x_0 := solve(g = f(\cdot, \frac{c+vcd}{2}), [a, b], \delta)$
  $g = subgradient(f, (x_0, \frac{c+d}{2}))$
  if g[1] > 0 **then**
    $Q := [a, b] \times [c, \frac{c+d}{2}]$
  else
    $Q := [a, b] \times [\frac{c+d}{2}, d]$
  end if
  $y_0 := solve(g = f(\frac{a+b}{2}, \cdot), [c, d], \delta)$
  $g := subgradient(f, (\frac{a+b}{2}), y_0)$
  if g[0] > 0 **then**
    $Q := [a, \frac{a+b}{2}] \times [c, d]$
  else
    $Q := [\frac{a+b}{2}, b] \times [c, d]$
  end if
  if StopRec() == False **then**
    Method(f, Q)
  end ifreturn $(\frac{a+b}{2}, \frac{c+d}{2})$
end function

3 Algorithm correctness

Let's $\mathbf{x}_0$ is solution of the task on segment, Q_1 is choosed rectangle, Q_2 is not choosed rectangle.

Lemma 3.1. *If the optimization task on segment is solved with zero error and the f is convex and differentiable at a point-solution, rectangle with solution of initial task was choosed correct.*

Proof. From sub-gradient definition, $\mathbf{x}^* \in \{x|(\mathbf{g}(\mathbf{x}_0), \mathbf{x}_0 - \mathbf{x}^*) \geq 0)\}$. Lemma's statement follows from it and a fact that the first (or the second for vertical segment) gradient's component in point $\mathbf{x}_0$ is zero. □

Theorem 3.1. *Let's the f has continuous derivative on the square. Then there is a neighbourhood of a solution of optimization task on segment such as a choice of rectangle will not change if one use any point from the neighbourhood.*

Proof. Let's consider a case when we work with horizontal segment. Case with vertical segment is considered analogously. Then we are interesting in $f'_y(x_0, y_0)$. If $\mathbf{x}_0$ is not solution of initial task, then $f'_y(x_0, y_0) \neq 0$.

From a continuity of the derivative:

$$\lim_{\delta \to 0} f'_y(x_0 + \delta, y_0) = f'_y(x_0, y_0)$$

Therefore,

$$\exists \delta_0 : \forall \mathbf{x}_\delta \in B_{\delta_0}(x_0) \times y_0 \Rightarrow \text{sign}(f'_y(\mathbf{x}_\delta)) = \text{sign}(f'_y(\mathbf{x}_0))$$

From it and lemma 4.1 theorem's statement follows.

□

The method does not work for all convex functions even for zero error on segment. There is an example of non-smooth convex problem in [2] when this method can not converge to the solution more accurately then one constant.

4 Error's Value

Let's find possible value of error's value, i.e. let's find a number δ_0 such as if a distance between a solution on a segment and its approximation is less δ_0 then rectangle for the segment is selected correctly (selected rectangle includes global solution on square). Rectangles are defined correctly for a horizontal optimization task, if:

$$\forall \delta : |\delta| < \delta_0 \Rightarrow f'_y(\mathbf{x}_0) f'_y(x_0 + \delta, y_0) > 0 \tag{1}$$

Analogically, for a vertical segment:

$$\forall \delta : |\delta| < \delta_0 \Rightarrow f'_x(\mathbf{x}_0) f'_x(x_0, y_0 + \delta) > 0 \tag{2}$$

Theorem 4.1. *Let function f be convex and has L-Lipschitz continuous gradient and a point $\boldsymbol{x}_0$ is a solution of optimization's task on a current segment.*

The current segment is horizontal and $\exists M > 0 :\Rightarrow |f'_y(\boldsymbol{x}_0)| \geq M$ or the current segment is vertical and $\exists M > 0 :\Rightarrow |f'_x(\boldsymbol{x}_0)| \geq M$. Then rectangle is defined correctly if the possible value of error is less than $\frac{M}{L}$.

Proof. Condition (1) is met if there is a derivative $f'_y(x_0 + \delta, y_0)$ in a neighbourhood of $f'_y(\mathbf{x}_0)$ with radius $\left|f'_y(\mathbf{x}_0)\right|$:

$$\left|f'_y(\mathbf{x}_0) - f'_y(x_0 + \delta, y_0)\right| < \left|f'_y(\mathbf{x}_0)\right|$$

The L-Lipschitz continuity gives following inequality:

$$\left|f'_y(\mathbf{x}_0) - f'_y(x_0 + \delta, y_0)\right| \leq L|\delta|$$

Therefore the following possible value is sufficient to select rectangle correctly:

$$\delta_0 < \frac{M}{L} \leq \frac{\left|f'_y(\mathbf{x}_0)\right|}{L}$$

Statement for vertical segment is proved similarly. □

In written above theorem the estimate needs some lower bound for derivatibe in point-solution on segment and this task can be hard in practise. Using other interpretation of the condition (1) one can take more useful result.

Theorem 4.2. *Let function f be convex and has L-Lipschitz continuous gradient and a point $\boldsymbol{x}_0$ is a solution of optimization's task on a current segment.*

The current segment is horizontal and $M = |f'_y(\boldsymbol{x}_{current})|$ or the current segment is vertical and $M = |f'_y(\boldsymbol{x}_{current})|$. Then rectangle is defined correctly if a distance between $\boldsymbol{x}_{cur}$ and accurate solution on segment is less than $\frac{M}{L}$.

Proof. Condition (1) is met if there is a derivative $f'_y(x_0 + \delta, y_0)$ in a neighbourhood of $f'_y(\mathbf{x}_0)$ with radius $\left|f'_y(\mathbf{x}_{cur})\right|$:

$$\left|f'_y(\mathbf{x}_0) - f'_y(\mathbf{x}_{cur})\right| < \left|f'_y(\mathbf{x}_{cur})\right|$$

The L-Lipschitz continuity gives following inequality:

$$\left|f'_y(\mathbf{x}_0) - f'_y(\mathbf{x}_{cur})\right| \leq L|\delta|$$

Therefore the following possible value is sufficient to select rectangle correctly:

$$\delta_0 < \frac{M}{L} \leq \frac{\left|f'_y(\mathbf{x}_0)\right|}{L}$$

Statement for vertical segment is proved similarly. □

Theorems 4.1 and 4.2 give stop conditions in the case when gradient in point-solution on segment and close points is large. But what should we do if gradient in this point is small?

Theorem 4.3. *Let f be convex and has L-Lipschitz continuous gradient. Then for accuracy on function ϵ following condition in point $\boldsymbol{x}$ is sufficient:*

$$\|\nabla f(\boldsymbol{x})\| \leq \frac{\epsilon}{a\sqrt{2}},$$

where a is size of current square.

Proof. Let's consider following inequality for convex functions (see prove in [3]):

$$f(\mathbf{x}^*) - f(\mathbf{x}) \geq (\nabla f(\mathbf{x}), \mathbf{x}^* - \mathbf{x})$$

Using Cauchy–Bunyakovsky–Schwarz inequality one has following inequality

$$\begin{aligned} f(\mathbf{x}) - f(\mathbf{x}^*) \leq -(\nabla f(\mathbf{x}), \mathbf{x}^* - \mathbf{x}) \leq \\ \leq \|\nabla f(\mathbf{x})\| \|\mathbf{x}^* - \mathbf{x}\| \leq \|\nabla f(\mathbf{x})\| a\sqrt{2} \end{aligned}$$

This inequality proves theorem's statement. □

Let's make a couple of remarks.

Firstly, we can replace the Lipschitz condition on the square by the Lipschitz condition on the segments in the theorems 4.1 and 4.2.

Secondly, we can use in this theorems Lipschitz constants not for gradients but for partitial derivative on segments. Its proves are obvious enough and almost repeat proves for this theorem.

5 Number of iterations

Below proved estimates are correct if each iterations was correct, i.e. after each iteration there is the solution in the selected rectangle. It is important condition

Theorem 5.1. *If function f is convex and L_f-Lipschitz continuous, then for to solve initial task with accuracy ϵ on function one should make following iteration's numbers:*

$$N = \left\lceil \log_2 \frac{\sqrt{2} L_f a}{\epsilon} \right\rceil \tag{3}$$

where a is a size of the initial square Q.

Proof. The square's size equals to $\frac{a}{2^k}$ on k^{th} iteration. If each irerations was correct then

$$\|\mathbf{x}_k - \mathbf{x}^*\| \leq \frac{a}{2^k}\sqrt{2}$$

Using the fact that function is Lipschitz continuous we have estimate 6 □

There are functions which estimates from written above theorem are very accurate for.

Example 5. Let's consider following task with positive constant A:

$$\min \left\{ A(x+y) | Q = [0,1]^2 \right\}$$

If one take a center of a current solution as approximate solution one have value $\frac{A}{2^N}$ after N iterations. Therefore, for accuracy ϵ one has to $\lceil \log_2 \frac{A}{\epsilon} \rceil$. For this function $L_f = 2A$. Therefore, estimate (6) is accurate for such tasks with little error that not more one iteration.

But any convex function is locally Lipschitz continuous at all $x \in \operatorname{int} Q$. Therefore, we have following theorem.

Theorem 5.2. *If function f is convex and a solution $x^* \in \operatorname{int} Q$, then for to solve initial task with accuracy ϵ on function one has to take a center of a current square as approximate solution and make following iteration's numbers:*

$$N = \left\lceil \log_2 \max \left\{ \frac{a}{\epsilon_0(\boldsymbol{x}^*)}, \frac{\sqrt{2} L_f a}{\epsilon} \right\} \right\rceil \tag{4}$$

where a is a size of the initial square Q, $\epsilon_0(x^*)$ is size of neighbourhood of x^* which f is L_f-Lipschitz continuous in, $\Delta f = f(x_0) - f(x^*)$, x_0 is a center of square Q.

We can improve written above estimates if to add new conditions:

Theorem 5.3. *Let function f be convex and has L-Lipschitz continuous gradient.*

If solution is a internal point, then for to solve initial task with accuracy ϵ on function one should make following iteration's numbers:

$$N = \left\lceil \frac{1}{2} \log_2 \frac{La^2}{\epsilon} \right\rceil \tag{5}$$

where a is a size of the initial square Q.

Proof. For all convex functions there is following inequality (one may find proof in [3]):

$$f(\mathbf{x}) - f(\mathbf{x}^*) - (f'(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^*) \leq \frac{L}{2}\|\mathbf{x} - \mathbf{x}^*\|^2$$

If $\mathbf{x}^*$ is a solution and an internal point, then $f'(\mathbf{x}^*) = 0$:

$$f(\mathbf{x}) - f(\mathbf{x}^*) \leq \frac{L}{2}\|\mathbf{x} - \mathbf{x}^*\|^2$$

After N iterations we have the estimate:

$$f(\mathbf{x}) - f(\mathbf{x}^*) \leq L\left(\frac{a}{2^N}\right)^2$$

Using it we have estimate (5). □

Estimate 6 works in all cases and better then estimate 5 when following condition is met:

$$\frac{2L_f^2}{L_g} \leq \epsilon,$$

где L_f, L_g is Lipschitz constants for function and for gradient. On the other hand, the estimate 5 works if point with zero gradient exists in the square.

6 Halving Cube Method

This Halving Square Method is optimization method for convex problem on square. But there is obvious generalization for convex problems in high dimensional spaces on n-dimensional hypercube. Let the its faces are parallel to axes. Let's describe it:

1. Take $(n-1)$-dimensional hyperplane in hypercube that includes hypercube center and is parallel to one of its face.

2. To solve optimization problem on this set with accaracy that is met to condition from theorem 4.2.

3. Calculate gradient in this point and select rectangle which antigradient look into.

4. Repeat the third step n times for each hyperplane in hypercube that includes hypercube center and is parallel to one of its face.

5. Repeat the previous step until required accuracy is achieved.

The proves for convergence for smooth function and for strategy repeat proves in the case of 2-dimensional space. Moreover, obvious enough that the following theorems are correct when on the fourth algorithm step rectangle was selected correctly.

Theorem 6.1. *If function f is convex and L_f-Lipschitz continuous, then for to solve initial task with accuracy ϵ on function one should make following iteration's numbers:*

$$N = \left\lceil \log_2 \frac{\sqrt{n} L_f a}{\epsilon} \right\rceil \tag{6}$$

where a is a size of the initial square Q.

Theorem 6.2. *Let function f be convex and has L-Lipschitz continuous gradient.*

If solution is a internal point, then for to solve initial task with accuracy ϵ on function one should make following iteration's numbers:

$$N = \left\lceil \frac{1}{2} \log_2 \frac{nLa^2}{2\epsilon} \right\rceil \tag{7}$$

where a is a size of the initial square Q.

Note the hyperplane in hypercube in the first step is $(n-1)$-dimensional hypercube. And we can use for it this method too. So, we solve the convex problem on n-dimensional hypercube recursively.

But this method has two essential disadvanteges for enough high dimensional spaces. Fisrtly, the required iterations number depend on logarithm of dimension. Of course, it is essential only for very high dimensional problems. Secondly, recursive algorithm will be extra slow because each the complexity of each iteration for this method is factotial of dimension.

We don't expect inspired result for each modification and we will not test it.

7 The Dual Problems

This method was created to solve dual task. Namely, following task:

$$-\phi(\lambda_1, \lambda_2) \to \min_{\lambda \geq 0}, \tag{8}$$

$$\text{where } \phi = \min_{\mathbf{x}} \left(f(\mathbf{x}) + \lambda_1 g_1(\mathbf{x}) + \lambda_2 g_2(\mathbf{x})\right) \tag{9}$$

In the section we will discuss how transform this task to task of the task of minimization on square, what is derivative and lipschitz constants and how to calculate its fastly.

7.1 Function Parameters

Fitstly, let's transform this task to the task of minimization on square. According to [1] (see ex. 4.1), we can add following restraint for the dual variables:

$$\|\lambda\|_1 \leq \lambda_{\max} = \frac{1}{\gamma}\left(f(\overline{\mathbf{x}}) - \min_{\mathbf{x}} f(\mathbf{x})\right), \tag{10}$$

$$\text{where } \overline{\mathbf{x}} : g_i(\overline{\mathbf{x}}) < 0, \gamma = \min_i g_i(\overline{\mathbf{x}}) \tag{11}$$

And we understand that there is the dual task's solution in a square $Q = [0, \lambda_{\max}]^2$. And we have following optimization task:

$$-\phi(\lambda_1, \lambda_2) \to \min_{\lambda \in Q} \tag{12}$$

Secondly, the gradient of function ϕ will be calculated according well-known Demyanov-Danskin-Rubinov Theorem, see [5].

Theorem 7.1. *Let $\phi(\lambda) = \min_{x \in X} \Phi(x, \lambda)$ for all $\lambda \geq 0$, where Φ is a smooth convex function with respect to λ and $x(\lambda)$ is the only maximum point. Then*

$$\nabla\phi(\lambda) = F'_\lambda\left(x(\lambda), \lambda\right)$$

If theorem's conditions is met for our case then we have derivative value:

$$\phi'_{\lambda_k}(\lambda) = g_k\left(\mathbf{x}(\lambda)\right) \tag{13}$$

Additionally, we need a Lipschitz constant for gradient. In the work [6] there is following theorem:

Theorem 7.2. *Let $f(x)$ be a μ_f-strongly convex function, the function $g(x)$ satisfies the Lipschitz condition with a constant M_g. Then the function $\phi(\lambda) = \min_{\boldsymbol{x}}(f(\boldsymbol{x} + \lambda_1 g_1(\boldsymbol{x}) + \lambda_2 g_2(\boldsymbol{x}))$ defined in 18, where $x(\lambda) = \arg\min_x(f(x)+ \lambda g(x))$, has Lipschitz smooth gradient with constant $L_{\phi'} = \frac{M_g^2}{\mu_f}$*

This theorem can be proved easy for the 2-dimensional space. In this case g is a vector-function.

In the following subsection we can use following notations:

$$\Phi(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda_1 g_1(\mathbf{x}) + \lambda_2 g_2(\mathbf{x})$$
$$\mathbf{x}(\lambda) = \arg\min_{\mathbf{x}} \Phi(\mathbf{x}, \lambda)$$
$$\phi(\lambda) = \min_{\mathbf{x}} \Phi(\mathbf{x}, \lambda) = \Phi(\mathbf{x}(\lambda), \lambda)$$

The next subsections are devoted to efficient calculation of function's and derivative's values.

7.2 Calcularting $\mathbf{x}(\lambda)$

We should have solution approximation on a segment for the one iteration. We will search it through the following dichotomy method: take a center of current segment and select such subsegment as antigradient looks at it. Obviously, that for to make one step on segment or on square we need the signum of derivative approximation matches with the derivative signum.

Let λ is a point which we want calculate derivative for the step on a segment or on the square. One should to calculate the derivative with respect to λ_k for to define what rectangle must be selected. If we used its approximation it is important that signum of approximation ϕ'_{λ_k} and of $\widetilde{\phi'_{\lambda_k}}$ are match. We have following sufficient condition for it:

$$|\phi'_{\lambda_k} - \widetilde{\phi'_{\lambda_k}}| \leq |\widetilde{\phi'_{\lambda_k}}|$$

$$|g_k(\mathbf{x}(\lambda)) - g_k(\mathbf{x})| \leq |g_k(\mathbf{x})|$$

If g_k is L_{g_k}-lipschitz continious function:

$$L_{g_k}\|\mathbf{x}(\lambda) - \mathbf{x}\| \leq |g_k(\mathbf{x})|$$

8 Other Inexact Methods

Our optimization method can solve the task of minimization function f on square when the function and its gradient can not be calculated accurately but there are other optimization method for such tasks. In each section one describes some of them and below there is experimental comparison of them with our method.

The fist method is Primal Gradient Method (PGM) with (δ, L, μ) oracle. There are proves in the [7] that this method converges to the solution with accuracy δ:

$$\min_k f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \frac{LR^2}{2} \exp\left(-k\frac{\mu}{L}\right) + \delta,$$

where $R = \|\mathbf{x}_0 - \mathbf{x}^*\|$ in our task. Moreover, in the [7] it is proved that for the function

$$f(\mathbf{x}) = \min_{\mathbf{u}} \left(\Psi(\mathbf{x}, \mathbf{u}) + \mathbf{u}^\top A\mathbf{x}\right)$$

there is following (δ, L, μ) oracle:

$$f_{\delta,L,\mu}(\mathbf{x}) = \Psi(\mathbf{x}, \mathbf{u}_\mathbf{x}) - \xi$$

$$g_{\delta,L,\mu}(\mathbf{x}) = A\mathbf{u}_\mathbf{x}$$

with parameters $\delta = 3\xi$, $L = \frac{2\lambda_{\max}(A^\top A)}{\mu(G)}$, $\mu = \frac{\lambda_{\min}(A^\top A)}{2L(G)}$ if $\mathbf{u}_\mathbf{x}$ is a solution approximation of $\mathbf{u}^*$ for current $\mathbf{x}$ with accuracy ξ on function.

The second method is Fast Gradient Method with (δ, L, μ) oracle. this method converges to the solution with accuracy δ:

$$\min_k f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \min\left(\frac{4LR^2}{k^2}, LR^2 \exp\left(-\frac{k}{2}\sqrt{\frac{\mu}{L}}\right)\right) + C_k\delta,$$

where $C_k = \min\left(\frac{k}{3} + \frac{12}{5}, 1 + \sqrt{\frac{L}{\mu}}\right)$. The method's description and proves for it is in the [7] too. This method has significantly better convergence rate for ill-conditioned problems.

The third having to be discussed method is inexact ellipsoid method. The ellipsoid method with ϵ-subgradient instead usual subgradient converges to a solution with accuracy ϵ:

$$\min_k f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \max_{\mathbf{x}\in Q} |f(\mathbf{x})| \exp\left(-\frac{k}{8}\right) + \delta$$

It is proved in [8]. Moreover, in [9] there is proved that for the function

$$f(\mathbf{x}) = \min_{\mathbf{u}} \Psi(\mathbf{x}, \mathbf{u})$$

the following statement is met:

$$\Psi(\mathbf{x}, \mathbf{u}_{\mathbf{x},\epsilon}) \in \partial_\epsilon f(\mathbf{x}),$$

if $\mathbf{u}_{\mathbf{x},\epsilon}$ is such point that $\Psi(\mathbf{x}, \mathbf{u}_{\mathbf{x},\epsilon}) - \min_{\mathbf{u}} \Psi(\mathbf{x}, \mathbf{u}) \leq \epsilon$.

Note that our method converge by the following way:

$$\min_k f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq C \exp(-k),$$

and it is the best theoretical convergence rate from all methods in this section.

9 Tests

In this section we show estimate on number iterations of practice. ALso there is comparison work time our new method with work time of other optimization methods such as inexact gradient descent and method ellipsoid[1]. All code was made in Anaconda 5.3.1 Python 3.6 (see cite [4])

9.1 Tests for iterations number

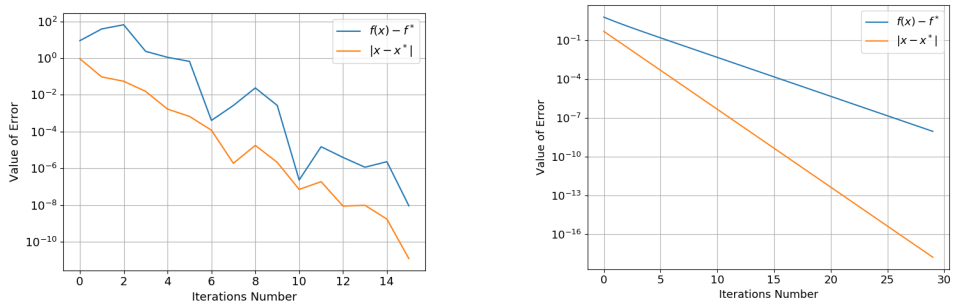
There is results of tests for iterations number. We consider task when global solution is internal point and external point. In 1 you can find values of estimates (see theorems 6, 5) for this expirements.

Estimate	Internal	External
Through L_f	40	30
Through $L_{f'}$	20	14

Table 1: Value for estimates through function's and gradient's lipschitz constant

Let's consider task of minimization semi-positive definite quadratic function with randomly generated parameters. We took such square that it involve global solution of this task.

[1]You can find all code in the repository [10]



(a) (b)

Figure 1: Test for iterations number (a) when the estimate 5 is reached; (b) when the estimate 5 is not reached.

We can see on 1(a) that the estimate 5 (see table 1) is reached in this case. But let's consider following task:

$$f(x, y) = (x + y)^2 + x^2, Q = [1, 2]^2, (x^*, y^*) = (1, 1)$$

$$\min_{(x,y)\in Q} f(x, y)$$

And we see on 1(b) that the estimate 5 (see table 1) is not reached in this case and in this case we should use estimate 6.

9.2 Comparison Of Stop Conditions

In current section we will compare different stop conditions for the solving task on separating segments. Fistly, let's define all stop conditions.

- Constant estimate (**Const**): $\delta \leq \frac{\epsilon}{2La\sqrt{5}\log_2 \frac{2Ma\sqrt{2}}{\epsilon}}$ from [2],
- Constant gradient estimate (**ConstGrad**): $\delta \leq \frac{M_{der}}{L}$ from 4.1,
- Estimate through current gradient (**CurGrad**): $\frac{|f'(\mathbf{x}_{cur})|}{L}$ from 4.2.

Obviously, that in practic we can not use the second strategy because to calculate this estimate for one segment is harder than to solve all task. This estimate is the improved the third estimate. And there is interesting question: how much better does the second strategy work than the third?

We will test on the simple quadratic functions. Parameters for this functions will be generated randomly. We will research dependence of work time on required accuracy ϵ. For each epsilon we will generate N tested functions and measure work time of each method. In our expiremetns N is equal to 1000 times. Results of experimetns you can see on the picture 9.2.

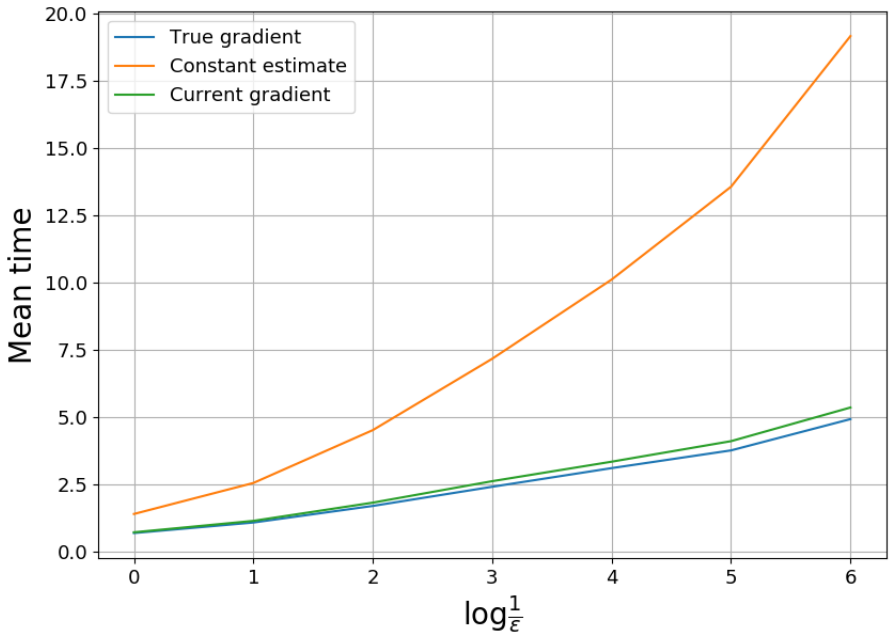


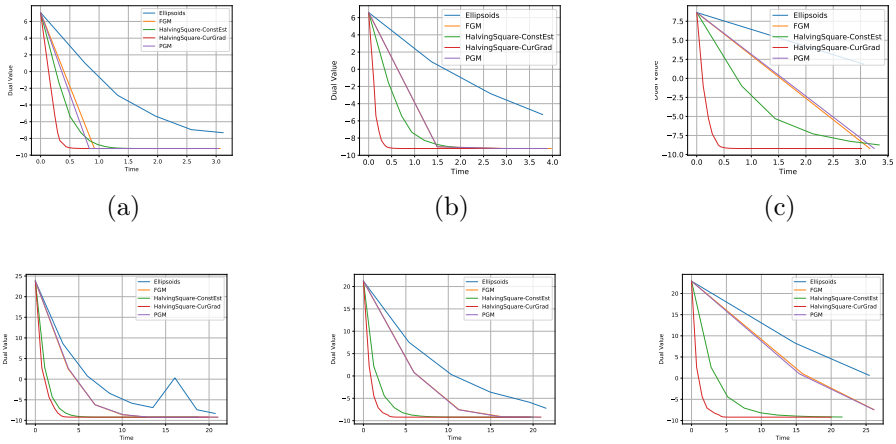
Figure 2: Comparison of different strategies for a segment

Firstly, we see that the first strategy is the slowest strategy from the three strategy. There is good explonation for it. The estimate for the iterations number is proportional to ϵ. The dichotomy method makes the steps number proportional to the logarithm of epsilon. Therefore, the number of iteration on segment for the first strategy is proportional to $(\ln \frac{1}{\epsilon})^2$. It decreases speed of this strategy. But for the saddle point problems calculating derivative that the second strategy needs is a hard and this fact can increase speed of the second strategy. But when we will test our method on such tasks that the second strategy is faster then the first too (see results on 3).

Secondly, we can see that the second and the third stratagies have similar results that's why we can state that the second estimate is good replacement

for the third. Of course, it is only for the simple tasks when calculating gradients and derivatives is fast procedure. Obviously, when to calculate gradient is the hard procedure we will not have such positive results.

9.3 Comparison With Other Methods



(a) (b) (c)

(d) (e) (f)

Figure 3: Comparison of different on inexact methods for task with different dimension N and for different required accuracy ϵ: (a) $N = 10, \epsilon = 10^{-1}$; (b) $N = 10, \epsilon = 10^{-3}$; (c) $N = 10, \epsilon = 10^{-10}$; (d) $N = 10, \epsilon = 10^{-1}$; (e) $N = 100, \epsilon = 10^{-3}$; (f) $N = 100, \epsilon = 10^{-10}$.

Let's compare our method with inexact ellipsoid method and primal gradient method with (δ, L, μ) oracle (see previous subsection 8) on a dual task.

For to find $\mathbf{x}(\lambda)$ we will use gradients descent. There are theoretical result that can help to manange distance $\|\mathbf{x}_k - \mathbf{x}^*\|$ from current to optimal points. In particular, there are following results:

- If f is a convex function with L-Lipschitz continious gradient then gradient descent with step $\alpha_k = \frac{1}{L}$ converges with speed

$$\|f(\mathbf{x}_k) - f(\mathbf{x}^*)\| \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|}{k+4}$$

- If f is a μ-strong convex function with L-Lipschitz continious gradient then gradient descent with step $\alpha_k = \frac{1}{L+\mu}$ converges with speed

$$\|f(\mathbf{x}_k) - f(\mathbf{x}^*)\| \leq \left(\frac{M-1}{M+1}\right)^k L\|\mathbf{x}_0 - \mathbf{x}^*\|,$$

  where $M = \frac{L}{\mu}$.

The proves for this statements one can find in many books of optimization, for example, in the book [9]. For all inexact method we will calculate $x(\lambda)$ with such accuracy as the method will converge to the solution with same for all methods accuracy ϵ

We consider following prime task:

$$f(\mathbf{x}) = \ln\left(1 + \sum_{k=1}^{m} e^{\mathbf{a}_k^\top \mathbf{x}}\right) + C\|\mathbf{x}\|_2^2 \to \min_{\mathbf{x}} \tag{14}$$

$$\text{s.t.}\, \mathbf{x} \in \mathbb{R} \tag{15}$$

$$g_k(\mathbf{x}) = \mathbf{b}_k^\top \mathbf{x} - R_k \leq 0, k = \overline{1,n} \tag{16}$$

$$\tag{17}$$

It is task of minimization the LogSumExp-function with $l2$-regularization. The regularization parameter C determines strong convexity of our task and in the tests one takes C equaling 1. The N is diminsionality of primal task and is determined for different tests below. Matrixes $A = \|\mathbf{a}_k\|_{k=1}^m \in \mathbb{R}^{N\times m}$ and $B = \|\mathbf{b}_k\|_{k=1}^n \in \mathbb{R}^{N\times n}$ are being generated randomly. The n is equal to dimensionality of dual task and in the current case is equal to 2. The m is equal to 10^4 in our experiments and has not changes.

We introduce the following notation:

$$\phi(\lambda_1, \lambda_2) = -\min_{\mathbf{x}} \left(f(\mathbf{x}) + \lambda_1 g_1(\mathbf{x}) + \lambda_2 g_2(\mathbf{x})\right) \tag{18}$$

In such notations the dual task for the task 14 looks like:

$$\phi(\lambda_1, \lambda_2) \to \min_{\lambda_1, \lambda_2} \tag{19}$$

$$\text{s.t}\, \lambda_1, \lambda_2 \geq 0 \tag{20}$$

Obviously, $\min_{\mathbf{x}} f(\mathbf{x}) \geq 0$. Therefore, according to 10 we can add following conditions on the dual variables:

$$|\lambda_k| \leq \lambda_{\max} = \frac{f(\overline{\mathbf{x}})}{\gamma}, k = 1, 2$$

And we have following task:

$$\phi(\lambda_1, \lambda_2) \to \min_{0 \leq \lambda_k \leq \lambda_{max}}$$

How to calculate function and derivatave value for such task was discussed in the section 7.

We can see on 3 the following results. Firstly, the halving square method with provided in this work strategy **CurGrad** are the fastest method in the most tests tests. This method can be slower than other inexact methods if dimensional of primal task is small or ϵ is big. In particular, this strategy is faster than strategy with constant estimate provided in [2]. It proves that provided by Nesterov method with strategy through gradient is the best method for to solve two dimensional dual task of minimization. Secondly, the gain of this strategy in comparison with other method is increase when the required ϵ decrease. This fact demonstrated important advantange of this strategy: it does not depend on required accuracy strongly. So, this method with constant estimate has strong dependity on it because there is this accuracy in the constant estimate, PGM and inexact ellipsoid method require that the $x(\lambda)$ is found with accuracy depended on ϵ. But halving square with ellipsoid method has not such dependety.

10 Conclusion

We discussed and proved that this method converges to the solution for smooth convex functon. Moreover, in the [2] there is conterexample when the problem is non-smooth and we can not to converge to the solution with the accuracy on function better than a constant.

After it we discussed different strategy for one-dimensional task. Two strategies were considered. The both suggest to use stop conditions that are met when the current approximation is "very near" to the segment's solution. The first compares the distance between them with derivative value in accurate segment's solution but the second compares it with derivative

value in approximation. In the experiment the first has a little better result but it can not be used for real task. The second strategy using derivative value in current approximation is significantly better then constant estimate and does not depend on required accuracy.

But all this strategy are good when the derivative value is high enough. But when the segment is near to the global solution this value will be small. Therefore one needs to make a lot of iterations on segment. For to avoid it we consider additional stop condition for global task on square when in current approximation the derivative value is near to zero.

The most steps of methods assumped that derivative can be calculated accurately. But the main method purpose is to solve dual problems and for it one can not usually calculate it so. That's why we consider different modifications of this method for to solve such problems. The important moment is we don't add some dependence on initial required accuracy in the modificated method.

Finally, we compared our method with new strategy for dual problem to prime LogSumExp problem with two linear constaints with our method with strategy using the constant estimate, primal gradient method and fast gradient method with (δ, L, μ) -oracle and with inexact ellipsoids methods. The Halving Square Method is the fastest of them for enough high dimension (more 100) and for enough high required solution ($1e-3$ and more).

References

[1] Gasnikov A. Universal gradient descent // MIPT — 2018, 240 p.

[2] Pasechnyk D.A., Stonyakin F.S. One method for minimization a convex Lipchitz continuous function of two variables on a fixed square // arXiv.org e-Print archive. 2018. – URL: https://arxiv.org/pdf/1812.10300.pdf

[3] Nesterov U.E. Methods of convex optimization // M.MCNMO — 2010, 262 p.

[4] Anaconda[site]. At available: https://www.anaconda.com

[5] Danskin, J.M.: The theory of Max-Min, with applications. J. SIAM Appl. Math.14(4) (1966)

[6] Fedor S. Stonyakin, Mohammad S. Alkousa, Alexander A. Titov,and Victoria V. Piskunova1 On Some Methods for Strongly Convex Optimization Problems with One Functional Constraint // ...

[7] Olivier Devolder Exactness, Inexactness and Stochasticityin First-Order Methods for Large-ScaleConvex Optimization // UCL — 2013,

[8] Need Reference To Book with Inexact Ellipsoids

[9] B.T. Polyak. The Introduction to Optimization // Moscow, Science - 1983

[10] Repository with code: https://github.com/ASEDOS999/Optimization-Halving-The-Square