ISSN 0005-1179 (print), ISSN 1608-3032 (online), Automation and Remote Control, 2025, Vol. 86, No. 3, pp. 266–279.

Robust Regression Modelling: Interior Point Methods, Simplex Method, Descent Along Nodal Straight Lines

O. A. Golovanov[*,a] **and A. N. Tyrsin**[**,***,b]

[*] *Institute of Economics, The Ural Branch of Russian Academy of Sciences, Ekaterinburg, Russia*
[**] *Ural Federal University named after the first President of Russia B. N. Yeltsin, Ekaterinburg, Russia*
[***] *Science and Engineering Center "Reliability and Safety of Large Systems and Machines", The Ural Branch of Russian Academy of Sciences, Ekaterinburg, Russia*
e-mail: [a] *golovanov.oa@uiec.ru,* [v] *at2001@yandex.ru*

Received March 14, 2024
Revised November 28, 2024
Accepted December 2, 2024

Abstract—Implementation of the least absolute deviations method for robust estimation of linear regression dependencies by means of interior point algorithms is considered. Two affine scaling interior point algorithms for robust regression estimation are implemented. A comparative analysis of these algorithms with simplex method and descent along nodal straight lines is carried out. Their computational complexity is found to be comparable to the simplex method, but they lose to the latter in terms of computation time. It is also found that the interior point algorithms significantly lose to the modified descent along nodal straight lines, both in terms of computational complexity and actual computation time. Examples of using interior point algorithms for practical problems are given.

Keywords: least absolute deviations method, linear regression, interior point method, computational efficiency

DOI: 10.31857/S0005117925030067

1. INTRODUCTION

Multiple linear regression is one of the widely used mathematical models in various fields of study. It takes the following form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\alpha} + \boldsymbol{\varepsilon},$$

where $\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$, $\mathbf{X} = (X_1, \ldots, X_m) = \begin{pmatrix} 1 & x_{12} & \cdots & x_{1m} \\ 1 & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n2} & \cdots & x_{nm} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_1^{\mathrm{T}} \\ \mathbf{x}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{x}_n^{\mathrm{T}} \end{pmatrix}$, $\boldsymbol{\alpha} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}$, $\boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_m \end{pmatrix}$.

Here $\mathbf{y}$ is the vector of dependent variable values Y; $\mathbf{X} = \{x_{ij}\}_{n\times m}$ is the matrix of explanatory variable values $X_1, \ldots, X_m$, $\mathbf{x}_i^{\mathrm{T}} = (1, x_{i2}, \ldots, x_{im})$; $\boldsymbol{\alpha}$ is the vector of unknown coefficients a_j of the regression equation, and $\boldsymbol{\varepsilon}$ is the vector of unobserved random deviations.

Traditional linear regression analysis dates back to the works of A.M. Legendre (1806) and C.F. Gauss (1809), in which they independently presented their versions of the least squares method (OLS), and the works of A.A. Markov a century later, in which the theoretical foundations were outlined. These premises define the requirements for variables X_i, parameters $\boldsymbol{\alpha}$, and random

deviations $\boldsymbol{\varepsilon}$, and allow for the investigation of properties and statistical content of the regression coefficient estimates [1].

1°) There are no constraints on the vector $\boldsymbol{\alpha}$, i.e., $\mathbf{A} = \mathbb{R}^m$, where $\mathbf{A}$ is the set of a priori values of the parameters $\boldsymbol{\alpha}$;

2°) The matrix $\mathbf{X}$ is deterministic, i.e., x_{ij} are not random variables;

3°) $\operatorname{rank}(\mathbf{X}) = m < n$;

4°) The vector $\boldsymbol{\varepsilon}$ is random, i.e., the vector $\mathbf{y}$ is also a random vector;

5°) The expected values are $\mathrm{E}(\varepsilon_i) = 0$, $i = 1, \ldots, n$, $\mathrm{E}(\varepsilon) = 0$;

6°) $\forall i \neq k \; cov(\varepsilon_i, \varepsilon_k) = 0$, $\forall i \; E(\varepsilon_i^2) = \sigma^2$, $i, k = 1, \ldots, n$, where σ^2 is the variance of deviations.

The last two assumptions concern the properties of random deviations. Since ε_i are unobservable and their properties are unknown a priori, the choice of method for estimating the parameters of $\boldsymbol{\alpha}$ is ambiguous. If the distribution of deviations $\boldsymbol{\varepsilon}$ does not depend on $\mathbf{X}$ and is normal, then the OLS provides best estimate of the $\tilde{\boldsymbol{\alpha}}$ regression coefficients [2].

However, it is not possible to assume a priori the normality of random deviations, as in many cases the actual distributions can differ significantly and have more elongated tails compared to the normal law, which reduces the accuracy of OLS estimates of the regression coefficients [3–6]. The situation is particularly critical for OLS when observations are "contaminated" by relatively rare outliers or misses that violate the 5° and 6° assumptions [2]. Contamination can occur, for example, in the development of degradation processes during the operation of mechanical and other loaded systems [7]. The random component of the vibration signal can be 10–20% contaminated by noise in the form of impulses, sharp level changes or correlation structure, leading to "heavier" tails of the distribution relative to the normal law [3].

In these cases, other methods are required to ensure stability of the estimates. The most well-known of these is the method of least absolute deviations (LAD) [8]. However, the LAD and other robust methods lose out to the performance of OLS in many applications, especially when analysing large samples or real-time data [3, 9]. This limits the applicability of regression analysis for these applications under conditions of stochastic heterogeneity of the data, when the OLS does not provide stability of model parameter estimation. Therefore, the task of increasing the computational efficiency of robust regression modelling algorithms is relevant.

The approach discussed is commonly referred to as a "passive" experimental model, where factors exist only in the form of controlled but not fully manageable input variables. The task of planning is reduced to the optimal organisation of information collection and selection of a method for processing the measurement results. The main disadvantages are that the range of factor changes is limited, and the influence of disturbing parameters may turn out to be more significant than the change in controlled factors.

It should be noted that if it is possible to influence the course of the process and to choose the factor levels in each test, then it is preferable to use an "active" experiment. The foundations of estimation theory were laid by R. Fisher in 1937 in his book The Design of Experiments. This extends the traditional regression model. In an active experiment, certain interventions are applied to the input of object under study, which are planned in advance according to some optimal criterion. In the last 50 years, schemes under the random nature of the regressor set $\mathbf{X}$ have been actively investigated. In the works of O.N. Granichin, B.T. Polyak, A.B. Tsybakov, M.C. Campi, Lei Guo, L. Ljung, and others, randomisation in the selection of regressors allowed for the formulation of faster algorithms and the study of their consistency when traditional assumptions about disturbances are not fulfilled [10–13]. It should be noted that active experiment allows to solve research problems faster and more efficiently, but it is not always realisable, is more complex and costly, and may interfere with the normal course of the technological process.

In the following section of the article, we will consider the passive experiment variant.

The task of estimating a multiple linear regression relationship on a data sample $(\mathbf{x}_i^{\mathrm{T}}, y_i)$, $i = 1, \ldots, n$, using LAD is of the form [14]:

$$Q(\mathbf{a}) = \sum_{i=1}^{n} \left| y_i - \sum_{j=1}^{m} a_i x_{ij} \right| \to \min_{\mathbf{a} \in \mathbb{R}^m}, \tag{1}$$

where $\mathbf{a} = (a_1, \ldots, a_m)^{\mathrm{T}}$ is the vector of unknown coefficient estimates a_j of the regression equation $Y = a_1 + a_2 X_2 + \cdots + a_m X_m + \varepsilon$.

One way to determine the parameters of $\mathbf{a}$ is to reduce (1) to a linear programming (LP) problem and solve it using the simplex method [14]. The computational efficiency of the simplex method for solving problem (1) was investigated in [15] and was found to be insufficient for dynamic applications.

An alternative to the simplex method for solving LP problems are interior point methods. These methods were originally used to solve nonlinear problems. In the context of LP problems, interior point methods explicitly or implicitly use a barrier on the feasible set in the form of a non-negative orthant. Unlike the simplex method, a sequence of points is generated for which the constraint inequalities are strictly satisfied.

The set of existing interior point methods can be conditionally divided into polynomial and affine scaling methods. The first algorithm based on affine scaling was proposed in 1967 by I.I. Dikin [16]. However, interior point methods gained widespread popularity in 1984 after the article by N. Karmarkar [17], in which a polynomial algorithm based on projective transformations was described. However, it was found that gradient-type optimisation methods with using affine scaling transformations were more efficient. The interior point methods based on projective transformations have [18, 19]

— complexity strongly depends (grows) on dimensionality of the problem being solved;
— iterations require higher computational costs compared to optimisation methods using affine scaling transformations;
— many polynomial algorithms are difficult to apply for a wide range of problems due to the necessity of finding an initial approximation.

This direction is being developed by many researchers, see works [20–23]. The history of interior point methods is described in [19].

The objective function $Q(\mathbf{a})$ in (1) is continuous, convex and bounded below, which guarantees the existence of a single minimum. However, problem (1) has specific features. First, the function $Q(\mathbf{a})$ has a number of kinks in the form of line segments. The walls of kinks represent convex linear faces. As the minimum is approached, geometry of the function $Q(\mathbf{a})$ deteriorates — the walls of kinks become more and more shallow and almost parallel, which complicates the convergence of algorithms near the minimum.

Secondly, the LP problems corresponding to (1) have high dimensionality.

No specific study of interior point methods for application to the class of LP problems corresponding to (1) has been found in the literature. Since many authors note that in practice interior point methods can compete with the simplex method [19], it seems reasonable to conduct a comparative analysis of the computational efficiency of interior point methods with other well-known exact methods for solving problem (1).

An algorithm is considered exact if it allows finding the global minimum of the function $Q(\mathbf{a})$ in a finite number of iterations. Calculations are performed with errors, and the technique of error-free calculations is very time-consuming, so we take as an exact solution the one that is computed with minimal computational errors. Among the exact ones we will include the LAD estimation algorithms based on solving the LP problem using the simplex method [7, 24, 25] and the descent along nodal straight lines algorithms [26, 27].

It should be noted that a number of other algorithms based on the LP problem are known, for example, algorithms that use at each iteration the fundamental operation of finding weighted medians over the local set of basis solutions [28–30]. Their complexity for solving problem (1) is comparable to the simplex method.

2. RESEARCH METHODS

Let us form an equivalent to (1) LP problem. We represent each residual $y_i - \sum_{j=1}^{m} a_j x_{ij}$ as

$$0 \leqslant \left| y_i - \sum_{j=1}^{m} a_j x_{ij} \right| \leqslant z_i, \quad i = 1, \ldots, n.$$

Hence we obtain the system

$$\begin{cases} z_i + \sum\limits_{j=1}^{m} a_j x_{ij} \geqslant y_i, & i = 1, \ldots, n, \\ z_i - \sum\limits_{j=1}^{m} a_j x_{ij} \geqslant -y_i, & i = 1, \ldots, n, \end{cases}$$

and, by denoting $a_j = a_j^{(1)} - a_j^{(2)}$, we can formulate the primal LP problem as

$$\begin{cases} \sum\limits_{i=1}^{n} z_i \to \min\limits_{a_j^{(k)}, z_i \in \mathbb{R}}, \\ z_i + \sum\limits_{j=1}^{m} (a_j^{(1)} - a_j^{(2)}) x_{ij} \geqslant y_i, & i = 1, \ldots, n, \\ z_i - \sum\limits_{j=1}^{m} (a_j^{(1)} - a_j^{(2)}) x_{ij} \geqslant -y_i, & i = 1, \ldots, n, \\ a_j^{(k)}, z_i \geqslant 0, & k = 1, 2, \end{cases} \tag{2}$$

or in matrix form

$$\begin{cases} \mathbf{b}^{\mathrm{T}} \tilde{\mathbf{y}} \to \min, \\ \mathbf{A}\tilde{\mathbf{y}} \geqslant \mathbf{C}, \\ \tilde{\mathbf{y}} \geqslant 0, \end{cases} \tag{3}$$

where $\mathbf{b}^{\mathrm{T}} = (\overbrace{1, 1, \ldots, 1}^{n}, \overbrace{0, \ldots, 0}^{2m})$ is a vector of size $1 \times (n + 2m)$, $\tilde{\mathbf{y}}$ is a vector of objective function values of size $(n + 2m) \times 1$, $\mathbf{C}$ is a vector of right-hand side constraints of size $2n \times 1$, $\mathbf{A}$ is a matrix of non-basic variable coefficients of size $2n \times (n + 2m)$.

Next, we consider interior point algorithms with affine scaling transformations, taking into account the aforementioned advantages over polynomial ones. Significant among them are I.I. Dikin [16] algorithm (algorithm **B**), which is essentially the progenitor of corresponding algorithms group, and its modification — V.I. Zorkaltsev [20] algorithm (algorithm **A**). Both algorithms combine the features of solving mutually dual LP problems:

$$\mathbf{c}^{\mathrm{T}}\mathbf{x} \to \min, \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geqslant 0;$$
$$\mathbf{b}^{\mathrm{T}}\mathbf{u} \to \max, \quad \mathbf{g}(\mathbf{u}) \geqslant 0, \quad \mathbf{g}(\mathbf{u}) = \mathbf{c} - \mathbf{A}^{\mathrm{T}}\mathbf{u},$$

where the matrix $\mathbf{A}$ of size $n \times m$, vectors $\mathbf{c} \in \mathbb{R}^m$, $\mathbf{b} \in \mathbb{R}^n$ are given. Vectors $\mathbf{x} \in \mathbb{R}^m$ and $\mathbf{u} \in \mathbb{R}^n$ are task variables.

2.1. Affine-Scaling Algorithm A

Let us describe the algorithm for solving the problem.

Step 1. At the kth iteration, we compute the vector of constraint-equality residuals, where $\mathbf{x}^{(0)}$ is any vector with positive components

$$\mathbf{r}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}, \quad k = 0, 1, 2, \ldots.$$

Step 2. Forming a matrix of weigh coefficients

$$\mathbf{D}_k = \operatorname{diag} \mathbf{d}^{(k)},$$

where $d_j^{(k)}$ is defined according to [20]. Then $d_j^{(k)} = (x_j^{(k)})^p$, $j = 1, \ldots, m$ for a given $p \geqslant 1$. Moreover, for $p > 1$ the additional condition $0 < \gamma \leqslant 2/(p+1)$ is required.

Step 3. Computing the vector of variables $\mathbf{u} \in \mathbb{R}^n$

$$\mathbf{u}^{(k)} = (\mathbf{A}\mathbf{D}_k\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{r}^{(k)} + \mathbf{A}\mathbf{D}_k\mathbf{c}).$$

Step 4. Finding the direction and step of the solution adjustment

$$\mathbf{s}^{(k)} = -\mathbf{D}_k\mathbf{g}(\mathbf{u}^{(k)}),$$
$$\lambda_k = \min\{1, \bar{\lambda_k}\}, \quad \text{if } \mathbf{r}^{(k)} \neq 0,$$
$$\lambda_k = \bar{\lambda_k}, \quad \text{if } \mathbf{r}^{(k)} = 0,$$

where $\bar{\lambda_k} = \gamma \min\{-x_j^{(k)}/s_j^{(k)} : s_j^{(k)} < 0\}$.

If $\mathbf{s}^{(k)} \geqslant 0$ when $\mathbf{r}^{(k)} \neq 0$, then we set $\lambda_k = 1$. Depending on the required accuracy when $\max_{j=1,\ldots,m} \left|s_j^{(k)}\right| \approx 0$ exit the algorithm, otherwise go to Step 5.

Step 5. Performing an iterative transition

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \lambda_k\mathbf{s}^{(k)},$$
$$\mathbf{r}^{(k+1)} = (1 - \lambda_k)\mathbf{r}^{(k)}.$$

Proceed to Step 2.

The exact solution is defined as the achievement of $\mathbf{s}^{(k)} = 0$, where each subsequent $\mathbf{x}^{(k)}$ will not change. However, in practice, only a certain value within an infinitesimally small neighbourhood of δ near zero is achieved, which varies depending on the development environment and the variables used in it. Thus, to avoid uncontrolled growth of the algorithm's runtime and to ensure uniformity regardless of the implementation tools, it is necessary to limit the accuracy of algorithm until a certain decimal place is reached.

Remark 1. When $\mathbf{r}^{(k)} = 0$, for each subsequent iteration, optimisation within the feasible region is performed

$$\mathbf{r}^{(k+1)} = 0, \quad \mathbf{c}^{\mathrm{T}}\mathbf{x}^{(k+1)} < \mathbf{c}^{\mathrm{T}}\mathbf{x}^{(k)}.$$

Let us consider a problem with equality constraints for a square matrix $\mathbf{A}$, where the vector of inequalities constraints $\mathbf{r}^{(k)} = 0$ reaches zero at the current iteration. Then, according to the formula of Step 3, we obtain $\mathbf{u}^{(k)} = (\mathbf{A}\mathbf{D}_k\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{A}\mathbf{D}_k\mathbf{c})$. Let us simplify the expression by replacing $\mathbf{A}\mathbf{D}_k = \mathbf{B}$, resulting in $\mathbf{u}^{(k)} = (\mathbf{B}\mathbf{A}^{\mathrm{T}})^{-1}(\mathbf{B}\mathbf{c})$. According to the properties of inverse matrix, we have $(\mathbf{B}\mathbf{A}^{\mathrm{T}})^{-1} = (\mathbf{A}^{\mathrm{T}})^{-1}\mathbf{B}^{-1}$. Let us expand the brackets and, using the associativity of matrix multiplication, compute the vector of variables $\mathbf{u} \in \mathbb{R}^n$, $\mathbf{u}^{(k)} = (\mathbf{A}^{\mathrm{T}})^{-1}\mathbf{B}^{-1}\mathbf{B}\mathbf{c} = (\mathbf{A}^{\mathrm{T}})^{-1}\mathbf{E}\mathbf{c} = (\mathbf{A}^{\mathrm{T}})^{-1}\mathbf{c}$. Then $\mathbf{g}(\mathbf{u}^{(k)}) = \mathbf{c} - \mathbf{A}^{\mathrm{T}}\mathbf{u}^{(k)} = \mathbf{c} - \mathbf{A}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}})^{-1}\mathbf{c} = \mathbf{c} - \mathbf{E}\mathbf{c} = 0$. Hence, at the current iteration $\mathbf{s}^{(k)} = -\mathbf{D}_k\mathbf{g}(\mathbf{u}^{(k)}) = 0$ and a solution of the linear algebraic equation system (SLAE) has been reached so no further optimisation is required.

Remark 2. The direction vector $\mathbf{s}^{(k)}$ is defined by solving the auxiliary problem

$$\mathbf{c}^{\mathrm{T}}\mathbf{s} + 1/2\,\mathbf{s}^{\mathrm{T}}\mathbf{D}_k^{-1}\mathbf{s} \to \min, \quad \mathbf{A}\mathbf{s}^{(k)} = \mathbf{r}^{(k)}.$$

Therefore, $\mathbf{A}\mathbf{s}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$, $\mathbf{s}^{(k)} = \mathbf{A}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)})$.

Thus, the special case for the matrix $\mathbf{A}$ with nonzero determinant can be solved by skipping Steps 2 and 3, reducing computational complexity and the impact of accumulated errors, which is especially useful when the analysed sample increases.

A number of computational experiments were carried out as part of the study, let us consider some of them.

Example 1. Let us solve the LP problem with a rectangular matrix $\mathbf{A}$: $F = \min(2x_1 - x_2)$; $x_1 + x_2 = 2$; $x_i \geqslant 0$, $i = 1, 2$ [31]. The optimal solution is $\mathbf{x} = (0; 2)$, $F_{\min} = -2$. As a initial point we will use $\mathbf{x}^{(0)} = (7; 1)$. Due to the rectangular form of the matrix, Remark 2 cannot be applied.

At the first iteration, the vector of inequality constraints will be equal to $\mathbf{r}^{(1)} = -6$. For $p = 2$ the vector of weight coefficients $\mathbf{d}^{(1)} = (49; 1)$ with $\gamma = 0.67$, whence $\mathbf{u}^{(1)} = 1.82$. Next, having defined the vector $\mathbf{g}(\mathbf{u}^{(1)}) = (0.18; -2.82)$, we will find the direction and step of the solution adjustment $\mathbf{s}^{(1)} = (-8.82; 2.82)$, $\lambda_1 = 0.53$. Thus, after the iterative transition we obtain $\mathbf{x}^{(2)} = (2.33; 2.49)$ and $\mathbf{r}^{(2)} = -2.83$. At the fifth iteration $\lambda_5 = 1$, so $\mathbf{r}^{(6)} = 0$. Since the number of model parameters exceeds the number of observations and inverse of the matrix $\mathbf{A}$ cannot be found due to it's determinant being zero, optimisation can be performed within the feasible region. Thus, $\lambda_6 = 3.5$ and $\mathbf{x}^{(6)} = (0.02; 1.98)$. Continuing the optimisation, we obtain the solution of Example 1, which is approximately $x_1 \approx 0, x_2 \approx 2$ with $F_{\min} \approx -2$ at the given accuracy of 10^{-8}, which corresponds to the global minimum of the problem. The solution does not change depending on the initial point.

Example 2. Let us expand the matrix to a square matrix: $F = \min(x_1 + 3x_2 + 2x_3)$; $5x_1 + 4x_2 + 7x_3 = 3$; $6x_1 + 3x_2 + 2x_3 = 2$; $x_1 + 2x_2 + 3x_3 = 1$; $x_i \geqslant 0$, $i = 1, 2, 3$. The minimum will be equal to $F_{\min} = 0.75$ with $\mathbf{x} = (1/4; 0; 1/4)$. We will set the initial point as $\mathbf{x}^{(0)} = (1; 1; 1)$. Since the matrix $\mathbf{A}$ is square and its inverse can be calculated, we will use Remark 2: $\mathbf{r}^{(1)} = (-13; -9; -5)$; $\mathbf{s}^{(1)} = (-0.75; -1; -0.75)$; $\lambda_1 = 0.67$. Then $\mathbf{x}^{(2)} = (0.5; 0.33; 0.5)$ and $\mathbf{r}^{(2)} = (-4.33; -3; -1.67)$. The vector of equality constraints $\mathbf{r}$ became zero on the 12th iteration, with solution $\mathbf{x}^{(12)} \approx (0.25; 0; 0.25)$ and $F_{\min} \approx 0.75$. Indeed, since the matrix $\mathbf{A}$ is invertible, $\mathbf{s}^{(12)} = 0$ and the solution of SLAE has been reached, no further optimisation is required.

Example 3. Let us modify the problem so that its global minimum is not equal to the solution of SLAE by introducing inequality constraints in Example 2: $F = \min(x_1 + 3x_2 - 2x_3)$; $5x_1 + 4x_2 + 7x_3 \leqslant 3$; $6x_1 + 3x_2 + 2x_3 \leqslant 2$; $x_1 + 2x_2 + 3x_3 \leqslant 1$; $x_i \geqslant 0$, $i = 1, 2, 3$. In this case, the minimum will be reached at $\mathbf{x} = (0; 0; 1/3)$ and will be equal to $F_{\min} = -2/3$. To take into account the inequality constraints, we will extend the matrix $\mathbf{A}$ with an identity matrix of size 3×3. As a result of the algorithm's operation, with a given accuracy 10^{-8} the solution $\mathbf{x}^{(24)} \approx (0; 0; 0.3)$ and $F_{\min} \approx -0.66$ was found after 24 iterations, which corresponds to the global minimum.

Statement 1. *The computational complexity of the primal affine scaling algorithm A for solving the given problem* (2) *will be* $O(n^{3,1})$.

The results indicate that the algorithm is capable of formally reaching an exact solution in a finite number of iterations. However, due to the exponential growth of the algorithm running time as the desired accuracy approaches zero, its application for a series of experiments with a sufficiently large number of observations ($n \geqslant 50$) becomes problematic.

2.2. Affine-Scaling Algorithm B

The solution of dual LP problem will be the vector $\mathbf{u}$, that satisfies the system of equations [16]

$$\sum_{t=1}^{n} b_{st}u_t = d_s,$$

where $b_{st} = \sum_{j=1}^{m} x_j^2 a_{sj}a_{tj}$, $d_s = \sum_{j=1}^{m} x_j^2 c_j a_{sj}$, $s = 1, \ldots, n$.

Let us assume

$$\mathbf{\Phi}(\mathbf{x}) = \sum_{j=1}^{m} x_j^2 \left(\sum_{i=1}^{n} a_{ij}u_i(\mathbf{x}) - c_j \right)^2, \quad s_j(\mathbf{x}) = x_j^2 \left(\sum_{i=1}^{n} a_{ij}u_i(\mathbf{x}) - c_j \right).$$

The solution algorithm is as follows. Let $x_j^{(0)} > 0$. Then $x_j^{(k+1)} = x_j^{(k)} + \lambda_k s_j(\mathbf{x}^{(k)})$, where $\lambda_k = 1/\sqrt{\mathbf{\Phi}(\mathbf{x}^{(k)})}$ at $j = 1, \ldots, m$. The computation check at each iteration is based on the condition fulfilment [31]

$$\sum_{j=1}^{m} c_j \sigma_j^{(k)} \delta_j^{(k)} = -\mathbf{\Phi}(\mathbf{x}^{(k)}), \tag{4}$$

where $\sigma_j^{(k)} = (x_j^{(k)})^2$, $\delta_j^{(k)} = \sum_{i=1}^{n} a_{ij}u_i^{(k)} - c_j$ at $j = 1, \ldots, m$.

The solutions of Examples 1–3 using the described method are exact, regardless of the initial point $\mathbf{x}^{(0)} > 0$. Let us consider an additional example.

Example 4. $F = \min(3x_1 + 2x_2 + x_3)$; $x_2 + x_3 \geqslant 4$; $2x_1 + x_2 + 2x_3 \geqslant 6$; $2x_1 - x_2 + 2x_3 \geqslant 2$; $x_i \geqslant 0$, $i = 1, 2, 3$. The exact solution is $F = 4$ in $\mathbf{x} = (0; 0; 4)$ for the primal LP problem and $\mathbf{y} = (1; 0; 0)$ for the dual. We will present the results for the first and last iterations of the algorithm. At the first iteration we obtain $\mathbf{B}^{(1)} = \begin{pmatrix} 3 & 3 & 1 \\ 3 & 10 & 7 \\ 1 & 7 & 10 \end{pmatrix}$, $\mathbf{d}^{(1)} = (3; 10; 6)$, whence $\mathbf{u}^{(1)} = (-0.12; 1.19; -0.22)$ and $\mathbf{\Phi}(\mathbf{x}^{(1)}) = 3.78$. The calculation with a accuracy of 10^{-8} was completed in 15 iterations with the results: $\mathbf{B}^{(15)} = \begin{pmatrix} 5 & 8 & 8 \\ 8 & 16 & 16 \\ 8 & 16 & 20 \end{pmatrix}$, $\mathbf{d}^{(15)} = (4; 8; 8)$, whence $\mathbf{u}^{(15)} = (0; 0.5; 0)$ and $\mathbf{\Phi}(\mathbf{x}^{(15)}) = 4.5 \times 10^{-8}$ with the solution $F_{\max} \approx 3$, which does not correspond to the exact solution, although condition (4) was satisfied at each iteration.

As a result of a series of computational experiments related to solving the given problem (2), it was found that the solution using algorithm tends to zero. This may be a consequence of the failure to satisfy the condition in [32], according to which all inequality constraints must be satisfied in strict form. Nevertheless, determining dependence of the average computation time on the sample size is impossible due to the lack of method convergence.

Statement 2. *The computational complexity of one iteration of the primal affine scaling algorithm B for solving the given problem* (2) *will be* $O(n^3)$*, which is comparable to algorithm A considering the number of iterations.*

The gradient descent along nodal straight lines algorithm is described in [26]. It consists of the following. Let us consider an $m$dimensional Euclidean space $\mathbb{R}^m$ with standard orthonormalised basis $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$, where $\mathbf{e}_k^{\mathrm{T}} = (\overbrace{0, \ldots, 0}^{k-1}, 1, \overbrace{0, \ldots, 0}^{m-k})$. Each non-degenerate observation $(\mathbf{x}_i^{\mathrm{T}}, y_i) =$

$(1, x_{i2}, \ldots, x_{im}, y_i), i = 1, \ldots, n$, forms in $\mathbb{R}^m$ a hyperplane $\Omega_i : y_i - \mathbf{a}^{\mathrm{T}}\mathbf{x}_i = 0$ in the orthogonal coordinate system $Oa_1 ... a_m$. The intersection of m independent hyperplanes forms a nodal point

$$\mathbf{u} = \cap_{s \in M} \Omega_S, \quad M = \{k_1, \ldots, k_m\}, \quad k_1 < k_2 < \cdots < k_m, k_l \in \{1, \ldots, n\}.$$

The intersection of $(m-1)$ independent hyperplanes forms a nodal straight line

$$l_{(k_1, \ldots, k_{m-1})} : \cap \Omega_i, \quad i \in \{k_1, \ldots, k_{m-1}\}, \quad k_l \in \{1, \ldots, n\}.$$

The solution to problem (1) is always at a nodal point. The algorithm performs a descent from an arbitrary initial nodal point along nodal straight lines. Any nodal point is the basic solution of the LP problem. The advantage over the simplex method lies in the fact that transition along nodal line is more efficient, because, unlike the simplex method, at each step of the transformation only n points located on the corresponding nodal line are involved, rather than all the data. In addition, at each step, the descent is performed to the point with minimum value among all m nodal lines intersecting it.

To reduce computational cost during the descent, analysis is not performed on objective function values, but on its derivatives along the direction of nodal line. In the modified version of the algorithm, the initial nodal point (initial approximation) is determined on a subset of the sample and calculations of the objective function values at the minima of the nodal lines are excluded [27].

3. DISCUSSION OF RESULTS

A comparison between the interior point algorithms and the simplex method shows that they are at least equally efficient in terms of computational complexity. Thus, the affine scaling algorithms A and B, considering the number of iterations, have a complexity of $O(n^{3,1})$, while the solution of the primal and dual LP problems using the simplex method has a complexities of $O(n^{3,2}m^{0,2})$ and $O(n^3 m^{0,5})$, respectively [15]. However, when $n \gg m$, they, are still significantly inferior to the modified gradient descent, which has a complexity of $O(n^{1,5}m^{1,8})$ [27].

The main difference between the gradient descent along nodal straight lines and the algorithms based on solving LP problems using the simplex method and interior point algorithms is that it operates only with the initial data until the global minimum is found. This guarantees convergence to the exact solution regardless of the number of iterations and simplicity of the method implementation. In each subsequent iteration, the latter uses information from the previous one contained in the simplex table or vectors respectively, which can lead to biased estimation due to the accumulation of computational errors. Nevertheless, considering this fact in the implementation, one can achieve a negligible number of deviations, although this also increases the computational cost.

Comparison of algorithms based on computational complexity estimation in *Big O* notation is necessary but insufficient. Its simplicity leads to neglecting constants, multiple minor summands, and algorithm memory consumption. Therefore, there may be situations where two algorithms with the same *Big O* have significantly different computation times, or conversely, algorithms with different *Big O* have the same computation time.

3.1. Comparative Analysis of Algorithms on Model Data

For a more comprehensive evaluation, we will use the Monte Carlo method and compare the execution times of the algorithms for 1000 experiments over $m = 2, 3, \ldots, 7$ and $n = 50, 100, \ldots, 500$. Since the goal is to investigate the computational efficiency of the algorithms rather than accuracy of the LAD-based regression model estimation, we use the standard normal distribution of random errors ε in the generated data samples. Generation was performed using the built-in functions of the

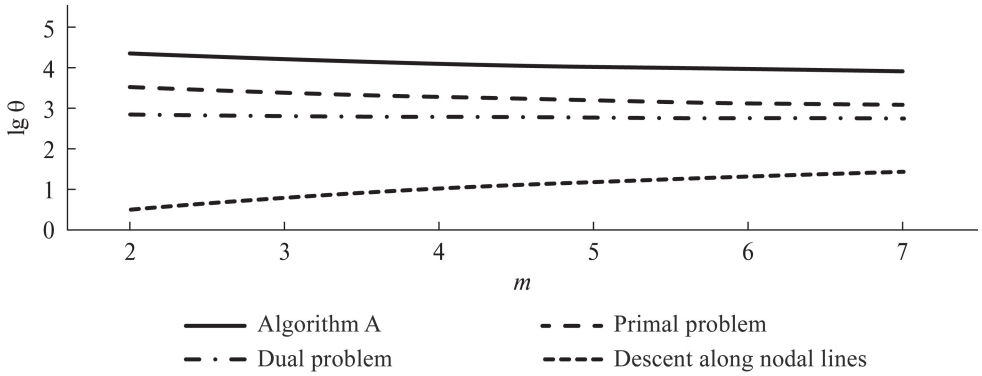

Fig. 1. Decimal logarithms of the ratios of computation times using LAD algorithms to the computation time of OLS when $n = 300$.

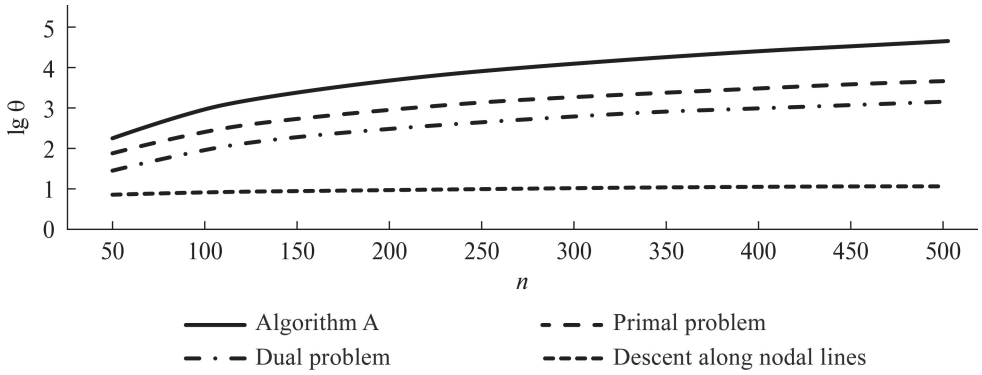

Fig. 2. Decimal logarithms of the ratios of computation times using LAD algorithms to the computation time of OLS when $m = 4$.

C++ programming language in the Microsoft Visual Studio 2019 environment. The computational experiments will be conducted on a Dell G5 5587 laptop with a 6-core i7-8750H processor with a clock frequency of up to 4.1 GHz. The accuracy for algorithm A is set to $\delta = 10^{-5}$.

As a result of determining the dependence of the algorithms' execution time on the number of observations and model parameters for the affine scaling algorithm A, we obtain $t_1 = 0.0001 \times n^{3.4}$ with a coefficient of determination $R^2 = 0.96$. As noted above, algorithm B is not applicable to solving the given problem (2), and therefore it will be excluded from further comparison. The dependencies for solving the primal and dual LP problems using the simplex method, as well as for the modified gradient descent, are as follows $t_2 = 0.0004 \times m^{0.001} n^{2.8}$, $t_3 = 0.0001 \times m^{0.6} n^{2.7}$, $t_4 = 0.00059 \times m^{2.51} n^{1.21}$ [27]. For the sake of clarity, Figs. 1 and 2 present graphs of decimal logarithms of execution time ratios for the considered algorithms to the computation time using OLS.

As a result, it can be stated that the affine scaling algorithm A is an order of magnitude slower than the simplex method and several orders of magnitude slower than the gradient descent.

3.2. Comparative Analysis of Algorithms on Practical Examples

Let us now consider three practical examples of regression modelling on real data. As the exact solution of problem (1), the results obtained using the brute-force algorithm for enumerating all nodal points were used in all the examples [15]. The results of calculation using the simplex method and the gradient descent algorithm along nodal straight lines coincided with the brute-force algorithm in all three examples. Thus, the simplex method and the gradient descent algorithm found exact solutions $\mathbf{a}^*$ of problem (1).

Example 5. Let us determine the model parameters of the average economic damage caused by fires in municipal districts (MD) of the Sverdlovsk region for the year 2012 on basis of the data provided in the [33]. The model is defined as $\hat{Y} = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4$, where $\hat{Y}$ is the average forecast damage from fires in the current year, mln rubles; $X_1 = 1$; X_2 is the number of buildings and structures in the MD, thsd units; X_3 is the total length of roadways in the MD territory, km; X_4 is the annual fire damage from the previous year, mln rubles; **a** is the vector of unknown parameters of the model; $n = 58$. The model is statistically significant, with a coefficient of determination $R^2 = 0.68$. The calculation results are given in Table 1.

Table 1. Results of coefficient parameter calculations **a** for Example 5

Algorithm	a_1	a_2	a_3	a_4	$Q(\mathbf{a})$	s	d
Solution $\mathbf{a}^*$	–3.822	1.403	–0.053	0.802	638.43	—	—
A ($\delta = 10^{-3}$)	–5.127	1.118	–0.042	0.847	644.90	20.0%	100.0%
A ($\delta = 10^{-4}$)	–2.451	1.223	–0.065	0.800	642.69	17.8%	111.7%
A ($\delta = 10^{-5}$)	–4.170	1.538	–0.053	0.794	640.76	5.0%	123.7%
A ($\delta = 10^{-6}$)	–3.615	1.534	–0.060	0.793	639.32	7.1%	135.0%
A ($\delta = 10^{-7}$)	–3.91	1.471	–0.057	0.804	638.45	3.6%	194.0%
A ($\delta = 10^{-8}$)	–3.855	1.428	–0.054	0.803	638.44	1.3%	206.7%

Example 6. To assess the relative performance of central processing units (CPU), data on their characteristics and relative performance are provided in [34]; $n = 209$. The machines represented a wide range of performance and manufacturers.

The prediction of relative CPU performance was carried out using the model $\hat{Y} = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4$, where $\hat{Y}$ is the estimated relative CPU performance; $X_1 = 1$; X_2 is the main memory size = (minimum main memory size + maximum main memory size)/2; X_3 is the cache memory size; X_4 is the channel bandwidth = (minimum number of channels + maximum number of channels)/(2 × machine cycle time); **a** is the vector of unknown model parameters. The model is statistically significant, with a coefficient of determination $R^2 = 0.89$. The calculation results are given in Table 2.

Table 2. Results of coefficient parameter calculations **a** for Example 6

Algorithm	a_1	a_2	a_3	a_4	$Q(\mathbf{a})$	s	d
Solution $\mathbf{a}^*$	–0.394	0.0072	0.5160	184.2	6190.5	—	—
A ($\delta = 10^{-3}$)	7.240	0.0034	0.8684	235.5	6538.2	521.3%	100.0%
A ($\delta = 10^{-4}$)	0.403	0.0079	0.4549	166.8	6233.9	58.3%	112.3%
A ($\delta = 10^{-5}$)	–4.661	0.0055	0.7103	196.7	6227.7	337.6%	122.7%
A ($\delta = 10^{-6}$)	–1.716	0.0079	0.4730	173.1	6205.5	89.9%	167.5%
A ($\delta = 10^{-7}$)	1.520	0.0066	0.5205	195.1	6198.6	125.1%	192.0%
A ($\delta = 10^{-8}$)	1.459	0.0065	0.5394	196.7	6198.6	122.9%	208.2%

Example 7. Consider the task of modelling wheat yield in Kirkuk based on climatic and socio-economic indicators using data from 2020 to 2022 ($n = 23$) [35]. We have a regression model $\hat{Y} = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4$, where $\hat{Y}$ is an estimate of wheat yield, tons/ha; $X_1 = 1$; X_2 is the population of Kirkuk, mln ppl; X_3 is the per capita gross domestic product of Iraq at 2023 prices (to account for inflation), thsd USD; X_4 is the normalised vegetation index; X_5 is the surface pressure, kPa/10; X_6 is the wind speed at a distance of at least 10 metres, m/s; **a** is the vector of unknown parameters of the model. The model is statistically significant, with a coefficient of determination $R^2 = 0.81$. The results of calculations are given in Table 3.

Table 3. Results of coefficient parameter calculations **a** for Example 7

Algorithm	a_1	a_2	a_3	a_4	a_5	a_6	$Q(\mathbf{a})$	s	d
Solution $\mathbf{a}^*$	610.2	1.918	–0.132	2.526	–62.92	4.019	5.02	—	—
A ($\delta = 10^{-3}$)	134.8	3.366	–0.287	–5.294	–13.87	4.438	9.57	111.4%	100.0%
A ($\delta = 10^{-4}$)	1292.8	2.891	–0.248	1.590	–133.1	–1.828	7.20	90.6%	113.8%
A ($\delta = 10^{-5}$)	584.7	3.209	–0.393	–1.360	–60.30	8.597	6.84	90.0%	124.1%
A ($\delta = 10^{-6}$)	593.0	3.274	–0.360	–1.288	–61.15	6.882	6.67	78.4%	137.9%
A ($\delta = 10^{-7}$)	192.1	2.938	–0.268	–0.632	–19.86	5.056	6.11	73.8%	213.8%
A ($\delta = 10^{-8}$)	675.5	2.217	–0.194	1.832	–69.57	0.274	5.71	34.0%	237.9%

In Tables 1–3, the following are denoted: s is the mean value of the absolute relative errors in calculation of the **a** coefficients (in %); d is the ratio of calculation time of algorithm A with accuracy δ to the calculation time with accuracy $\delta_0 = 10^{-3}$.

Conclusions from Examples 5–7:
— in the considered accuracy range with decreasing δ the calculation time increases by 2–2.5 times;
— the convergence speed of algorithm A to the minimum of the objective function $Q(\mathbf{a})$ decreases with increasing problem dimensionality and sample size, with the growth of m being more critical.

4. CONCLUSION

Using the affine scaling algorithms of V.I. Zorkaltsev (algorithm A) and I.I. Dikin (algorithm B) as examples, the efficiency of interior point methods for LAD estimation of regression models was analysed.

The analysis of the interior point algorithms A and B showed that when solving problem (1):
— their computational complexity is comparable to the simplex method, but they are slower in terms of computation time,
— they significantly (by more than an order of magnitude) lose to the modified descent along nodal straight lines both in terms of computational complexity and actual computation time,
— the convergence speed of algorithm A to the minimum of the objective function $Q(\mathbf{a})$ decreases with increasing problem dimensionality and data sample size, with increasing dimensionality being more critical
— increasing the accuracy raises runtime of algorithm A, but it is less critical compared to the dimensionality of the problem and the size of the analysed data sample.

The results indicate that algorithm A, without taking into account computational errors, is capable of reaching an exact solution in a finite number of iterations. However, the dependence of computation time on the given accuracy, problem dimensionality, and data sample size limits the scope of its application for solving problems of the form (1) to the values $\delta \geqslant 10^{-8}$, $m \leqslant 4$, $n \leqslant 100$.

The time loss of algorithm A compared to the simplex method, despite having approximately the same complexity, is caused by the presence of a set of almost parallel very small edges near the minimum of the objective function, which complicates and reduces the efficiency of using barriers.

Only two algorithms for implementing interior point methods have been considered in this article. However, since there is still no information about cardinal (by an order of magnitude or more) improvement in the actual performance of modern algorithms, it can be argued that descent along nodal straight lines is more efficient for calculating the parameters of linear regression models from experimental data than interior point methods.

APPENDIX

Proof of Statement 1. The computational complexity of matrix multiplication $(m \times n)$ by $(n \times l)$ is $O(mnl)$, and finding the inverse using the Gauss–Jordan method for a square matrix of dimension

$(n \times n)$ is $O(n^3)$. Let $n_1 = 2n$; $n_2 = 3n + 2m$, then the matrix dimensions for the given problem (2) will be: $\mathbf{A}$ — $(n_1 \times n_2)$; $\mathbf{x}$ — $(n_2 \times 1)$; $\mathbf{b}$ — $(n_1 \times 1)$; $\mathbf{c}$ — $(n_2 \times 1)$. Note that when $n \gg m$ in *Big O* notation, the weight of the operations number n_1 and n_2 will be comparable. Let us define the complexity of each step of the algorithm for solving the problem (1) reduced to a LP:

Step 1. $\mathbf{A}\mathbf{x}^{(k)}$ is the matrix multiplication of size $(n_1 \times n_2)$ by $(n_2 \times 1)$ and will be computed in $O(n_1 n_2)$ operations. Together with the subtraction operation, the total complexity of this step will be $O(n_1 + n_1 n_2)$ or $P_1 = O(n_1 n_2)$.

Step 2. The vector $\mathbf{x}$ is raised to a power p and a diagonal matrix is formed from it. Since raising to $p = 2$ is equivalent to multiplying the number by itself, its complexity is $P_2 = O(n_2)$, which is the complexity of step due to the fact that the diagonal matrix is set by initialisation from the resulting vector.

Step 3. Let us break it down step by step:

1) the matrix multiplication $\mathbf{A}\mathbf{D}_k\mathbf{A}^{\mathrm{T}}$ will be calculated in $(n_1 n_2^2 + n_1^2 n_2)$ operations;

2) the inverse of found matrix will be determined in (n_1^3) operations;

3) the expression $(\mathbf{r}^{(k)} + \mathbf{A}\mathbf{D}_k\mathbf{c})$ in $(n_1 n_2^2 + n_1 n_2 + n_1)$ operations;

4) multiplication of the resulting matrices will be performed in (n_1^2) operations.

The total computational complexity will be $O(n_1^3 + 2n_1 n_2^2 + n_1^2 n_2 + n_1^2 + n_1 n_2 + n_1)$ or $P_3 = O(n_1^3)$ for $n \gg m$.

Step 4. The vector $\mathbf{g}(\mathbf{u}^{(k)})$ is computed in $(n_1 n_2 + n_2)$ operations. Then the total complexity will be $O(n_2^2 + n_1 n_2 + n_2)$ or $P_4 = O(n_2^2)$.

Step 5. The iterative transition is performed in $P_5 = O(n_2)$.

Steps 2–5 are performed in a loop until a stop point is reached. Thus, the total computational complexity of the algorithm will be:

$$\begin{aligned} P &= P_1 + (P_2 + P_3 + P_4 + P_5) \times \{\text{number of iterations}\} \\ &= n_1 n_2 + (n_2 + n_1^3 + n_2^2 + n_2) \times \{\text{number of iterations}\}, \end{aligned}$$

where the number of iterations is a constant depending on the given accuracy of the algorithm.

Using the Monte Carlo statistical testing method, it was found that the optimal specified solution accuracy for a standard normal distribution of random errors δ in the analysed sample is 10^{-5}, due to low accuracy at 10^{-3} and long computation time at 10^{-8}. Therefore, the algorithm stops when the maximum absolute value of the vector $\mathbf{s}$ is less than 10^{-5}, which allows maintaining an acceptable level of solution accuracy regardless of the analysed sample size.

For 100 iterations with $m = 2, 3, \ldots, 7$ and $n = 50, 100, \ldots, 500$, on average, the algorithm will find the solution in 5 iterations with a deviation from the exact solution by 14.3%. To take into account the possible influence of the iterations number on the computational complexity of the algorithm, regardless of the given solution accuracy (when n significantly exceeds the number of iterations), let us increase the degree at n

$$O(P) = O(n_1^3 \times \{\text{number of iterations}\}) \leqslant O(n^{3,1}).$$

Statement 1 is proven.

Proof of Statement 2. Let us define the computational complexity of one iteration under the condition that the given problem (2) is solved — for determining the dual estimates the matrix $\mathbf{B}$ is computed in $O(4n_1^2 n_2)$ and the vector $\mathbf{d}$ is computed in $O(4n_1 n_2)$ operations, respectively.

Thus, the computation of the solution vector $\mathbf{u}$ is performed in $O(n_1^3 + 4n_1^2 n_2 + n_1^2 + 4n_1 n_2)$ or $O(n_1^3)$ operations. The vector $\mathbf{s}(\mathbf{x})$ will be found in $O(2n_2(2n_1 + 1))$ or $O(n_1 n_2)$, while $\mathbf{\Phi}(\mathbf{x})$ will be found in $O(3n_2(2n_1 + 1))$ or $O(n_1 n_2)$ operations.

The overall computational complexity of one iteration will be $O(n_1^3)$ or $O(n^3)$ operations. Taking into account the possible number of iterations depending on the given solution accuracy, the method can indeed be considered comparable to algorithm A.

Statement 2 is proven.

REFERENCES

1. Greene, W.H., *Econometric Analysis: 8th ed.*, New York: Pearson, 2020.
2. Aivazyan, S.A., Yenyukov, I.S., and Meshalkin, L.D., *Prikladnaya statistika: issledovanie zavisimostei* (Applied Statistics: Study of Relationships), Moscow: Finansy i Statistika, 1985, 206 p.
3. Babkin, N.V., Musaev, A.A., and Makshanov, A.V., *Robastnye metody statisticheskogo analiza navigatsionnoi informatsii: obzor* (Robust Methods of Statistical Analysis of Navigation Information: Review), Chelpanov, I.B., Ed., Leningrad: TsNII "Rumb", 1985, 206 p.
4. Orlov, A.I., On the Requirements for Statistical Methods of Data Analysis (Generalizing Article), *Industrial Laboratory. Diagnostics of Materials*, 2023, vol. 89, no. 11, pp. 98–106. https://doi.org/10.26896/1028-6861-2023-89-11-98-106
5. Salls, D., Torres, J.R., Varghese, A.C., Patterson, J., and Pal, A., Statistical Characterization of Random Errors Present in Synchrophasor Measurements, *IEEE Power & Energy Society General Meeting (PESGM)*, Washington, DC, USA, 2021, pp. 1–5. https://doi.org/10.1109/PESGM46819.2021.9638135
6. Ives, A.R., Random Errors are Neither: On the Interpretation of Correlated Data, *Methods in Ecol. and Evol.*, 2022, vol. 13, no. 10, pp. 2092–2105. https://doi.org/10.1111/2041-210X.13971
7. Kolobov, A.B., *Vibrodiagnostika: teoriya i praktika* (Vibrodiagnostics: Theory and Practice), Moscow: Infra-Inzheneriya, 2019, 252 p.
8. Dodge, Y., *The Concise Encyclopedia of Statistics*, Springer, 2008, 616 p. https://doi.org/10.1007/978-0-387-32833-1
9. Akimov, P.A. and Matasov, A.I., An Iterative Algorithm for l_1-Norm Approximation in Dynamic Estimation Problems, *Autom. Remote Control*, 2015, vol. 76, no. 5, pp. 733–748. https://doi.org/10.1134/S000511791505001X
10. Granichin, O.N. and Polyak, B.T., *Randomizirovannye algoritmy otsenivaniya i optimizatsii pri pochti proizvol'nykh pomekhakh* (Randomized Algorithms for Estimation and Optimization in the Presence of Nearly Arbitrary Disturbances), Moscow: Nauka, 2003, 291 p.
11. Nazin, A.V., Nemirovsky, A.S., Tsybakov, A.B., and Juditsky, A.B., Algorithms of Robust Stochastic Optimization Based on Mirror Descent Method, *Autom. Remote Control*, 2019, vol. 80, no. 9, pp. 1607–1627. https://doi.org/10.1134/S0005117919090042
12. Guo, L. and Ljung, L., Performance Analysis of General Tracking Algorithms, *IEEE Trans. on Autom. Contr.*, 1995, vol. 40, no. 8, pp. 1388–1402. https://doi.org/10.1109/9.402230
13. Campi, M.C. and Weyer, E., Guaranteed Non-Asymptotic Confidence Regions in System Identification, *Automatica*, 2005, vol. 41, no. 10, pp. 1751–1764. https://doi.org/10.1016/j.automatica.2005.05.005
14. Mudrov, V.I. and Kushko, V.L., *Metody obrabotki izmerenii: kvazipravdopodobnye otsenki* (Measurement Processing Techniques: Quasi-Plausible Estimates), Moscow: URSS, 2022, 304 p.
15. Golovanov, O.A. and Tyrsin, A.N, Descent Along Nodal Straight Lines and Simplex Algorithm: Two Variants of Regression Analysis Based on The Least Absolute Deviation Method, *Industrial Laboratory. Diagnostics of Materials*, 2024, vol. 90, no. 5, pp. 79–87. https://doi.org/10.26896/1028-6861-2024-90-5-79-87
16. Dikin, I.I., Iterative Solution of Linear and Quadratic Programming Problems, *Doklady AN SSSR*, 1967, vol. 174, no. 4, pp. 747–748.
17. Karmarkar, N., A New Polynomial-Time Algorithm for Linear Programming, *Combinatorica*, 1984, vol. 4, no. 4, pp. 373–395. https://doi.org/10.1145/800057.808695
18. Bayer, D.A. and Lagarias, J.C., The Nonlinear Geometry of Linear Programming I: Affine and Projective Scaling Trajectories, *Trans. of the Amer. Math. Soc.*, 1989, vol. 314, pp. 499–526. https://doi.org/ 10.1090/S0002-9947-1989-1005525-6

19. Vorontsova, E.A., Hildebrandt, R.F., Gasnikov, A.V., and Stonyakin, F.S., *Vypuklaya optimizatsiya* (Convex Optimization), Moscow: MIPT, 2021, 364 p.

20. Zorkal'tsev, V.I., Interior Point Method: History and Prospects, *Comput. Math. and Math. Phys.*, 2019, vol. 59, no. 10, pp. 1597–1612. https://doi.org/10.1134/S0965542519100178.

21. Nesterov, Yu., and Nemirovski, A., *Interior-Point Polynomial Algorithms in Convex Programming*, SIAM Studies in Applied Math., Philadelphia: SIAM, 1994, vol. 13, p. 414.

22. Lee, Y.T. and Yue, M.-C., Universal Barrier is n-Self-Concordant, *E-print*, 2018. https://doi.org/10.48550/arXiv.1809.03011

23. Bubeck, S. and Eldan, R., The Entropic Barrier: Exponential Families, Log-Concave Geometry, and Self-Concordance, *Math. of Oper. Research*, 2019, vol. 44, no. 1, pp. 264–276. https://doi.org/10.1287/ moor.2017.0923

24. Panyukov, A.V. and Mezal, Ya.A., Parametric Identification of Quasilinear Difference Equation, *Bull. SUSU Ser. Math. Mech. Phys.*, 2019, vol. 11, no. 4, pp. 32–38. https://doi.org/10.14529/mmph190404

25. Barrodale, I. and Roberts, F.D.K., An Improved Algorithm for Discrete L_1 Linear Approximation, *SIAM J. Numer. Anal.*, 1973, vol. 10, pp. 839–848.

26. Tyrsin, A.N., Algorithms for Descent Along Nodal Straight Lines in The Problem of Estimating Regression Equations Using the Least Absolute Deviations Method, *Industrial Laboratory. Diagnostics of Materials*, 2021, vol. 87, no. 5, pp. 68–75. https://doi.org/10.26896/1028-6861-2021-87-5-68-75

27. Golovanov, O.A. and Tyrsin, A.N., Modification of Least Absolute Deviations Method Based on Gradient Descent Along Nodal Lines, *MMTT*, 2023, no. 11, pp. 43–46. https://doi.org/10.52348/2712-8873_MMTT_2023_11_43

28. Wesolowsky, G.O., A New Descent Algorithm for the Least Absolute Value Regression Problem, *Commun. in Stat., Simul. and Comput.*, 1981, vol. B10, no. 5, pp. 479–491.

29. Li, Y. and Arce, G.R., A Maximum Likelihood Approach to Least Absolute Deviation Regression, *EURASIP J. Advan. Signal Proc.*, 2004, vol. 12, pp. 1762–1769. https://doi.org/10.1155/ S1110865704401139

30. Wei Xue, Wensheng Zhang, and Gaohang Yu., Least Absolute Deviations Learning of Multiple Tasks, *J. Indust. Management Optim.*, 2018, no. 14(2), pp. 719–729. https://doi.org/10.3934/jimo.2017071

31. Zorkaltsev, V.I. and Dikin, I.I., *Iterativnoe reshenie zadach matematicheskogo programmirovaniya (algoritmy metoda vnutrennikh tochek)* (Iterative Solution of Mathematical Programming Problems (Algorithms of the Interior Point Method)), Novosibirsk: Nauka, 1980, 144 p.

32. Filatov, A.Yu., *Razvitie algoritmov vnutrennikh tochek i ikh prilozhenie k sistemam neravenstv* (Development of Interior Point Algorithms and Their Application to Systems of Inequalities), PhD Thesis, Cand. Sci. (Phys. Math.), Irkutsk, 2001, 123 p.

33. Tuzhikov, E.N., *Metodika otsenki effektivnosti deyatel'nosti organov mestnogo samoupravleniya po obespecheniyu pervichnykh mer pozharnoi bezopasnosti (na primere Sverdlovskoi oblasti)* (Methodology for Assessing the Effectiveness of Local Government Agencies in Ensuring Primary Fire Safety Measures (Example of Sverdlovsk Region)), PhD Thesis, Cand. Sci. (Tech.), Ekaterinburg, 2014, 186 p.

34. Ein-Dor, P. and Feldmesser, J., Attributes of the Performance of Central Processing Units: A Relative Performance Prediction Model, *Commun. of the ACM*, 1987, vol. 30, no. 4, pp. 308–317.

35. Tyrsin, A.N. and Hasan, A.M.X., Analysis of Factors Influencing Grain Yields in Iraq Using the Example of Kirkuk, *MMTT*, 2024, no. 8, pp. 81–87.

This paper was recommended for publication by A.A. Bobtsov, a member of the Editorial Board