

# Оптимизация, системный анализ и исследование операций

© 2025 г. О.А. ГОЛОВАНОВ (golovanov.oa@uiesc.ru)  
(Институт экономики Уральского отделения РАН, Екатеринбург),  
А.Н. ТЫРСИН, д-р техн. наук (at2001@yandex.ru)  
(Уральский федеральный университет  
им. первого Президента России Б.Н. Ельцина, Екатеринбург;  
Научно-инженерный центр  
«Надежность и ресурс больших систем и машин»  
Уральского отделения РАН, Екатеринбург)

## УСТОЙЧИВОЕ РЕГРЕССИОННОЕ МОДЕЛИРОВАНИЕ: МЕТОДЫ ВНУТРЕННЕЙ ТОЧКИ, СИМПЛЕКС-МЕТОД, СПУСК ПО УЗЛОВЫМ ПРЯМЫМ

Рассмотрен вопрос реализации метода наименьших модулей для устойчивого оценивания линейных регрессионных зависимостей с помощью алгоритмов внутренних точек. Реализованы два аффинно-масштабирующих алгоритма внутренних точек для устойчивого оценивания регрессии. Проведен сравнительный анализ этих алгоритмов с симплекс-методом и спуском по узловым прямым. Их вычислительная сложность оказалась сопоставимой с симплекс-методом, однако они проигрывают последнему по времени вычислений. Также установлено, что алгоритмы внутренней точки значительно проигрывают модифицированному спуску по узловым прямым как по вычислительной сложности, так и по фактическому времени вычислений. Приведены примеры использования алгоритмов внутренних точек для практических задач.

*Ключевые слова:* метод наименьших модулей, линейная регрессия, метод внутренней точки, вычислительная эффективность.

DOI: 10.31857/S0005231025030063, EDN: IEVGNK

### 1. Введение

Множественная линейная регрессия является одной из распространенных математических моделей в различных предметных областях. Она имеет вид

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

где  $\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$ ,  $\mathbf{X} = (X_1, \dots, X_m) = \begin{pmatrix} 1 & x_{12} & \cdots & x_{1m} \\ 1 & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n2} & \cdots & x_{nm} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{pmatrix}$ ,  $\boldsymbol{\alpha} =$

$$= \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}, \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_m \end{pmatrix}. \text{ Здесь } \mathbf{y} \text{ – вектор значений зависимой переменной } Y;$$

$\mathbf{X} = \{x_{ij}\}_{n \times m}$  – матрица значений объясняющих переменных  $X_1, \dots, X_m$ ,  $\mathbf{x}_i^T = (1, x_{i2}, \dots, x_{im})$ ;  $\boldsymbol{\alpha}$  – вектор неизвестных коэффициентов  $a_j$  уравнения регрессии,  $\boldsymbol{\varepsilon}$  – вектор ненаблюдаемых случайных отклонений.

Традиционный линейный регрессионный анализ берет начало с работ А.М. Лежандра (1806 г.) и К.Ф. Гаусса (1809 г.), в которых они независимо изложили свои версии метода наименьших квадратов (МНК), и трудами А.А. Маркова через 100 лет, в которых были изложены его теоретические обоснования. Эти предпосылки определяют требования к переменным  $X_i$ , параметрам  $\boldsymbol{\alpha}$  и случайным отклонениям  $\boldsymbol{\varepsilon}$  и позволяют исследовать свойства и статистическое содержание оценок коэффициентов регрессии [1]:

- 1°) На вектор  $\boldsymbol{\alpha}$  не наложено ограничений, т.е.  $\mathbf{A} = \mathbb{R}^m$ , где  $\mathbf{A}$  – множество априорных значений параметров  $\boldsymbol{\alpha}$ ;
- 2°) Матрица  $\mathbf{X}$  детерминирована, т.е.  $x_{ij}$  не являются случайными переменными;
- 3°)  $\text{rank}(\mathbf{X}) = m < n$ ;
- 4°) Вектор  $\boldsymbol{\varepsilon}$  – случайный, т.е. вектор  $\mathbf{y}$  – также случайный вектор;
- 5°) Математические ожидания  $E(\varepsilon_i) = 0$ ,  $i = 1, \dots, n$ ,  $E(\boldsymbol{\varepsilon}) = 0$ ;
- 6°)  $\forall i \neq k \text{ cov}(\varepsilon_i, \varepsilon_k) = 0$ ,  $\forall i E(\varepsilon_i^2) = \sigma^2$ ,  $i, k = 1, \dots, n$ ,  $\sigma^2$  – дисперсия отклонений.

Последние две предпосылки относятся к свойствам случайных отклонений. Поскольку  $\varepsilon_i$  ненаблюдаемы и их свойства априори неизвестны, то выбор метода оценивания параметров  $\boldsymbol{\alpha}$  неоднозначен. Если распределение отклонений  $\boldsymbol{\varepsilon}$  не зависит от  $\mathbf{X}$  и является нормальным, то МНК позволяет получить наилучшую оценку  $\tilde{\boldsymbol{\alpha}}$  коэффициентов регрессии [2].

Однако априори предполагать нормальность случайных отклонений нельзя, во многих случаях фактические распределения могут существенно отличаться и имеет более вытянутые хвосты по сравнению с нормальным законом, что снижает точность МНК-оценок коэффициентов регрессии [3–6]. Для МНК особенно критична ситуация, когда наблюдения «засорены» относительно редкими выбросами или промахами, нарушающими предпосылки 5° и 6° [2]. Засорение может возникнуть, например, при развитии деградиционных процессов в процессе эксплуатации механических и иных нагруженных систем [7]. Случайная компонента вибросигнала может быть на 10–20% засорена помехами в виде импульсов, резких изменений уровня или корреляционной структуры, приводящими к «утяжелению» хвостов распределения относительно нормального закона [3].

В этих случаях для обеспечения устойчивости оценок необходимы другие методы. Наиболее известным среди них является метод наименьших модулей (МНМ) [8]. Однако МНМ и другие устойчивые методы проигрывают в быст-

родействии МНК во многих приложениях, особенно когда приходится анализировать выборки большого размера, а также при анализе данных в режиме реального времени [3, 9]. Это ограничивает возможности использования регрессионного анализа для этих приложений в условиях стохастической неоднородности данных, когда МНК не обеспечивает устойчивости оценивания параметров моделей. Поэтому актуальна задача повышения вычислительной эффективности устойчивых алгоритмов регрессионного моделирования.

Рассмотренный подход обычно называют моделью «пассивного» эксперимента, при котором существуют только факторы в виде входных контролируемых, но неуправляемых переменных. Задача планирования сводится к оптимальной организации сбора информации и выбору метода обработки результатов измерений. Основные недостатки – диапазон изменений факторов ограничен, а влияние возмущающих параметров может оказаться более существенным, чем изменение контролируемых факторов.

Отметим, что если есть возможность воздействия на ход процесса и выбора в каждом опыте уровней факторов, то предпочтительнее использовать «активный» эксперимент. Основы теории оценивания заложил R. Fisher в 1937 г. в книге «The Design of Experiments». Это расширяет традиционную регрессионную модель. При активном эксперименте на вход исследуемого объекта подаются определенные воздействия, которые заранее планируются в соответствии с некоторым оптимальным критерием. В последние 50 лет активно исследовались схемы при случайной природе набора регрессоров  $\mathbf{X}$ . В работах О.Н. Граничина, Б.Т. Поляка, А.Б. Цыбакова, М.С. Campi, Lei Guo, L. Ljung и др. рандомизация при выборе регрессоров позволяла формулировать более быстрые алгоритмы и изучать их состоятельность при невыполнении традиционных предположений о помехах [10–13]. Отметим, что активный эксперимент позволяет быстрее и эффективнее решать задачи исследования, но не всегда реализуем, более сложен и затратен и может помешать нормальному ходу технологического процесса.

В статье далее будем рассматривать вариант пассивного эксперимента.

Задача оценивания множественной линейной регрессионной зависимости по выборке данных  $(\mathbf{x}_i^T, y_i)$ ,  $i = 1, \dots, n$ , с помощью МНМ имеет вид [14]:

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

где  $\mathbf{a} = (a_1, \dots, a_m)^T$  – вектор оценок неизвестных коэффициентов  $a_j$  уравнения регрессии  $Y = a_1 + a_2 X_2 + \dots + a_m X_m + \varepsilon$ .

Одним из путей определения параметров  $\mathbf{a}$  является сведение (1) к задаче линейного программирования (ЛП) и ее решение с помощью симплекс-метода [14]. Вычислительная эффективность симплекс-метода для решения задачи (1) была исследована в [15] и оказалась недостаточна для динамических приложений.

Для решения задач ЛП альтернативой симплекс-методу являются методы внутренней точки. Они вначале использовались для решения нелинейных задач. Применительно к задаче ЛП методы внутренней точки в явном или неявном виде используют барьер на допустимом множестве в форме неотрицательного ортанта. В отличие от симплекс-метода здесь генерируется последовательность точек, для которых строго выполняются ограничения-неравенства.

Совокупность существующих методов внутренних точек можно условно разделить на полиномиальные и аффинно-масштабируемые. Впервые алгоритм, основанный на аффинном масштабировании, был предложен в 1967 г. И.И. Дикиным [16]. Однако широкую популярность методы внутренней точки получили в 1984 г. после статьи Н. Кармаркара [17], в которой был описан полиномиальный алгоритм, основанный на проективных преобразованиях. Но было установлено, что оптимизационные методы градиентного типа с аффинно-масштабируемыми преобразованиями оказались более эффективными. У методов внутренних точек на основе проективных преобразований [18, 19]:

- сложность сильно зависит (растет) от размерности решаемой задачи;
- итерации требуют больших вычислительных затрат по сравнению с оптимизационными методами, использующими аффинно-масштабируемые преобразования;
- многие полиномиальные алгоритмы затруднительны в применении для широкого круга задач в связи с необходимостью поиска начального приближения.

Данное направление развивается многими исследователями, см. работы [20–23]. История методов внутренней точки описана в [19].

Целевая функция  $Q(\mathbf{a})$  в (1) является непрерывной, выпуклой и ограниченной снизу, что гарантирует наличие единственного минимума. Однако у задачи (1) есть специфические особенности. Во-первых, функция  $Q(\mathbf{a})$  имеет множество изломов в виде отрезков прямых. Стенки изломов представляют собой выпуклые линейные грани. При приближении к минимуму геометрия функции  $Q(\mathbf{a})$  ухудшается – стенки изломов становятся все более мелкими и почти параллельными, что затрудняет сходимость алгоритмов вблизи минимума.

Во-вторых, соответствующие (1) задачи ЛП имеют высокую размерность.

В литературе не обнаружено специального исследования методов внутренней точки применительно к классу задач ЛП, соответствующих (1). Поскольку многие авторы отмечают, что методы внутренних точек могут конкурировать на практике с симплекс-методом [19], представляется целесообразным провести сравнительный анализ вычислительной эффективности методов внутренней точки с другими известными точными методами именно для решения задачи (1).

Точным считаем алгоритм, позволяющий за конечное число итераций найти глобальный минимум функции  $Q(\mathbf{a})$ . Вычисления осуществляются с погрешностями, а техника безошибочных вычислений очень трудоемка, поэтому в качестве точного решения примем то, которое вычисляется с минимальными вычислительными погрешностями. К числу точных отнесем алгоритмы МНМ-оценивания, основанные на решении задачи (ЛП) симплекс-методом [7, 24, 25] и алгоритмы спуска по узловым прямым [26, 27].

Отметим, что известен еще ряд алгоритмов, основанных на задаче (ЛП), например, алгоритмы, которые на каждой итерации используют фундаментальную операцию нахождения взвешенных медиан по локальному множеству базисных решений [28–30]. Их сложность для решения задачи (1) сопоставима с симплекс-методом.

## 2. Методы исследования

Сформируем эквивалентную (1) задачу ЛП. Представим каждую невязку  $y_i - \sum_{j=1}^m a_j x_{ij}$  в виде

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

Отсюда получим систему

$$\begin{cases} z_i + \sum_{j=1}^m a_j x_{ij} \geq y_i, & i = 1, \dots, n, \\ z_i - \sum_{j=1}^m a_j x_{ij} \geq -y_i, & i = 1, \dots, n, \end{cases}$$

и, обозначив  $a_j = a_j^{(1)} - a_j^{(2)}$ , сформируем прямую задачу ЛП как

$$(2) \quad \begin{cases} \sum_{i=1}^n z_i \rightarrow \min_{a_j^{(k)}, z_i \in \mathbb{R}}, \\ z_i + \sum_{j=1}^m (a_j^{(1)} - a_j^{(2)}) x_{ij} \geq y_i, & i = 1, \dots, n, \\ z_i - \sum_{j=1}^m (a_j^{(1)} - a_j^{(2)}) x_{ij} \geq -y_i, & i = 1, \dots, n, \\ a_j^{(k)}, z_i \geq 0, & k = 1, 2, \end{cases}$$

или в матричном виде

$$(3) \quad \begin{cases} \mathbf{b}^T \tilde{\mathbf{y}} \rightarrow \min, \\ \mathbf{A} \tilde{\mathbf{y}} \geq \mathbf{C}, \\ \tilde{\mathbf{y}} \geq 0, \end{cases}$$

где  $\mathbf{b}^T = (\overbrace{1, 1, \dots, 1}^n, \overbrace{0, \dots, 0}^{2m})$  – вектор размерностью  $1 \times (n + 2m)$ ,  $\tilde{\mathbf{y}}$  – вектор значений целевой функции размерностью  $(n + 2m) \times 1$ ,  $\mathbf{C}$  – вектор правой части ограничений размерностью  $2n \times 1$ ,  $\mathbf{A}$  – матрица коэффициентов небазисных переменных размерностью  $2n \times (n + 2m)$ .

Далее рассмотрим алгоритмы внутренней точки с аффинно-масштабируемыми преобразованиями, учитывая указанные выше их преимущества перед полиномиальными. Знаковыми среди них являются алгоритм И.И. Дикина [16] (алгоритм  $\mathbf{B}$ ), фактически являющийся прародителем соответствующей группы алгоритмов, и его модификация – алгоритм В.И. Зоркальцева [20] (алгоритм  $\mathbf{A}$ ). Оба алгоритма совмещают в себе особенности решения взаимно двойственных задач ЛП:

$$\begin{aligned} \mathbf{c}^T \mathbf{x} &\rightarrow \min, \quad \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0; \\ \mathbf{b}^T \mathbf{u} &\rightarrow \max, \quad \mathbf{g}(\mathbf{u}) \geq 0, \quad \mathbf{g}(\mathbf{u}) = \mathbf{c} - \mathbf{A}^T \mathbf{u}, \end{aligned}$$

где заданы матрица  $\mathbf{A}$  размера  $n \times m$ , векторы  $\mathbf{c} \in \mathbb{R}^m$ ,  $\mathbf{b} \in \mathbb{R}^n$ . Векторы  $\mathbf{x} \in \mathbb{R}^m$  и  $\mathbf{u} \in \mathbb{R}^n$  являются переменными задач.

### 2.1. Аффинно-масштабирующий алгоритм $A$

Опишем алгоритм решения задачи.

*Шаг 1.* На  $k$ -й итерации вычисляем вектор невязок ограничений-равенств, здесь в качестве  $\mathbf{x}^{(0)}$  выступает любой вектор с положительными компонентами

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

*Шаг 2.* Формируем матрицу весовых коэффициентов

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

где  $d_j^{(k)}$  зададим согласно [20]. Тогда  $d_j^{(k)} = (x_j^{(k)})^p$ ,  $j = 1, \dots, m$  при заданном  $p \geq 1$ . Причем при  $p > 1$  требуется дополнительное условие  $0 < \gamma \leq 2/(p + 1)$ .

*Шаг 3.* Вычисляем вектор переменных  $\mathbf{u} \in \mathbb{R}^n$

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

*Шаг 4.* Находим направление и шаг корректировки решения

$$\begin{aligned} \mathbf{s}^{(k)} &= -\mathbf{D}_k\mathbf{g}(\mathbf{u}^{(k)}), \\ \lambda_k &= \min\{1, \bar{\lambda}_k\}, \quad \text{если } \mathbf{r}^{(k)} \neq 0, \\ \lambda_k &= \bar{\lambda}_k, \quad \text{если } \mathbf{r}^{(k)} = 0, \end{aligned}$$

где  $\bar{\lambda}_k = \gamma \min \left\{ -x_j^{(k)} / s_j^{(k)} : s_j^{(k)} < 0 \right\}$ .

Если  $\mathbf{s}^{(k)} \geq 0$  при  $\mathbf{r}^{(k)} \neq 0$ , то полагаем  $\lambda_k = 1$ . В зависимости от необходимой точности при  $\max_{j=1, \dots, m} |s_j^{(k)}| \approx 0$  выходим из алгоритма, иначе переходим к шагу 5.

*Шаг 5.* Проведем итеративный переход

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

Переход к шагу 2.

Точным решением будем называть достижение методом  $\mathbf{s}^{(k)} = 0$ , при котором каждый последующий  $\mathbf{x}^{(k)}$  не будет изменяться. Однако на практике достигается лишь некоторое значение в бесконечно малой окрестности  $\delta$  около нуля, варьирующейся в зависимости от среды разработки и используемых в ней переменных. Таким образом, чтобы избежать неконтролируемого роста времени работы алгоритма и приведения его к единообразию вне зависимости от средств реализации, необходимо ограничить точность алгоритма вплоть до достижения некоторого знака после запятой.

*Замечание 1.* При  $\mathbf{r}^{(k)} = 0$  для каждой последующей итерации происходит оптимизация в области допустимых решений

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

Пусть решается задача с ограничениями-равенствами для квадратной матрицы  $\mathbf{A}$ , достигшей на текущей итерации нуля для вектора ограничений-неравенств  $\mathbf{r}^{(k)} = 0$ . Тогда, согласно формуле шага 3, получим  $\mathbf{u}^{(k)} = (\mathbf{A} \mathbf{D}_k \mathbf{A}^T)^{-1} (\mathbf{A} \mathbf{D}_k \mathbf{c})$ . Упростим выражение, заменив  $\mathbf{A} \mathbf{D}_k = \mathbf{B}$ , в результате  $\mathbf{u}^{(k)} = (\mathbf{B} \mathbf{A}^T)^{-1} (\mathbf{B} \mathbf{c})$ . Согласно свойствам обратной матрицы имеем  $(\mathbf{B} \mathbf{A}^T)^{-1} = (\mathbf{A}^T)^{-1} \mathbf{B}^{-1}$ . Раскроем скобки и, используя ассоциативность умножения матриц, вычислим вектор переменных  $\mathbf{u} \in \mathbb{R}^n$ ,  $\mathbf{u}^{(k)} = (\mathbf{A}^T)^{-1} \mathbf{B}^{-1} \mathbf{B} \mathbf{c} = (\mathbf{A}^T)^{-1} \mathbf{E} \mathbf{c} = (\mathbf{A}^T)^{-1} \mathbf{c}$ . Тогда  $\mathbf{g}(\mathbf{u}^{(k)}) = \mathbf{c} - \mathbf{A}^T \mathbf{u}^{(k)} = \mathbf{c} - \mathbf{A}^T (\mathbf{A}^T)^{-1} \mathbf{c} = \mathbf{c} - \mathbf{E} \mathbf{c} = 0$ . Следовательно, на текущей итерации  $\mathbf{s}^{(k)} = -\mathbf{D}_k \mathbf{g}(\mathbf{u}^{(k)}) = 0$  и было достигнуто решение системы линейных алгебраических уравнений (СЛАУ), и дальнейшая оптимизация не требуется.

*Замечание 2.* Вектор направления  $\mathbf{s}^{(k)}$  определяем, решая вспомогательную задачу

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

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

Таким образом, частный случай для матрицы  $\mathbf{A}$  с ненулевым определителем можно решить, пропуская шаги 2 и 3, сократив вычислительную сложность и влияние накопительных ошибок, что особенно полезно при увеличении анализируемой выборки.

В рамках исследования был проведен ряд вычислительных экспериментов, рассмотрим некоторые из них.

*Пример 1.* Решим задачу ЛП с прямоугольной матрицей  $\mathbf{A}$ :  $F = \min(2x_1 - x_2)$ ;  $x_1 + x_2 = 2$ ;  $x_i \geq 0$ ,  $i = 1, 2$  [31]. Оптимальным решением является  $\mathbf{x} = (0; 2)$  с  $F_{\min} = -2$ . В качестве начальной точки будем использовать  $\mathbf{x}^{(0)} = (7; 1)$ . В силу прямоугольного вида матрицы нельзя воспользоваться замечанием 2.

На первой итерации вектор ограничений-неравенств будет равен  $\mathbf{r}^{(1)} = -6$ . При  $p = 2$  вектор весовых коэффициентов  $\mathbf{d}^{(1)} = (49; 1)$  с  $\gamma = 0,67$ , откуда  $\mathbf{u}^{(1)} = 1,82$ . Далее, определив вектор  $\mathbf{g}(\mathbf{u}^{(1)}) = (0,18; -2,82)$ , найдем направление и шаг корректировки решения  $\mathbf{s}^{(1)} = (-8,82; 2,82)$ ,  $\lambda_1 = 0,53$ . Таким образом, после итеративного перехода получим  $\mathbf{x}^{(2)} = (2,33; 2,49)$  и  $\mathbf{r}^{(2)} = -2,83$ . На пятой итерации  $\lambda_5 = 1$ , поэтому  $\mathbf{r}^{(6)} = 0$ . Поскольку число параметров модели больше числа наблюдений и обратная для матрицы  $\mathbf{A}$  не находится в силу равенства нулю ее определителя, возможно провести оптимизацию в области допустимых решений. Таким образом,  $\lambda_6 = 3,5$  и  $\mathbf{x}^{(6)} = (0,02; 1,98)$ . Продолжив оптимизацию, получим решение примера 1, которое будет примерно равно  $x_1 \approx 0$ ,  $x_2 \approx 2$  с  $F_{\min} \approx -2$  при заданной точности  $10^{-8}$ , что соответствует глобальному минимуму задачи. Решение не изменяется в зависимости от начальной точки решения.

*Пример 2.* Расширим матрицу до квадратной:  $F = \min(x_1 + 3x_2 + 2x_3)$ ;  $5x_1 + 4x_2 + 7x_3 = 3$ ;  $6x_1 + 3x_2 + 2x_3 = 2$ ;  $x_1 + 2x_2 + 3x_3 = 1$ ;  $x_i \geq 0$ ,  $i = 1, 2, 3$ . Минимум будет равен  $F_{\min} = 0,75$  при  $\mathbf{x} = (1/4; 0; 1/4)$ . Начальную точку зададим как  $\mathbf{x}^{(0)} = (1; 1; 1)$ . Поскольку матрица  $\mathbf{A}$  квадратная и можно вычислить ее обратную, воспользуемся замечанием 2:  $\mathbf{r}^{(1)} = (-13; -9; -5)$ ;  $\mathbf{s}^{(1)} = (-0,75; -1; -0,75)$ ;  $\lambda_1 = 0,67$ . Тогда  $\mathbf{x}^{(2)} = (0,5; 0,33; 0,5)$  и  $\mathbf{r}^{(2)} = (-4,33; -3; -1,67)$ . Вектор ограничений-равенств  $\mathbf{r}$  стал равен нулю на 12-й итерации, с решением  $\mathbf{x}^{(12)} \approx (0,25; 0; 0,25)$  и  $F_{\min} \approx 0,75$ . Действительно, так как матрица  $\mathbf{A}$  имеет обратную,  $\mathbf{s}^{(12)} = 0$  и было достигнуто решение СЛАУ, дальнейшая оптимизация не требуется.

*Пример 3.* Видоизменим задачу, чтобы ее глобальный минимум не был равен решению СЛАУ, для чего введем в пример 2 ограничения-неравенства:  $F = \min(x_1 + 3x_2 - 2x_3)$ ;  $5x_1 + 4x_2 + 7x_3 \leq 3$ ;  $6x_1 + 3x_2 + 2x_3 \leq 2$ ;  $x_1 + 2x_2 + 3x_3 \leq 1$ ;  $x_i \geq 0$ ,  $i = 1, 2, 3$ . В таком случае минимум будет достигнут в  $\mathbf{x} = (0; 0; 1/3)$  и будет равен  $F_{\min} = -2/3$ . Для учета ограничений-неравенств расширим матрицу  $\mathbf{A}$  единичной матрицей размером  $3 \times 3$ . В результате работы алгоритма при заданной точности в  $10^{-8}$  за 24 итерации было найдено решение  $\mathbf{x}^{(24)} \approx (0; 0; 0,3)$  и  $F_{\min} \approx -0,66$ , что соответствует глобальному минимуму.

*Утверждение 1.* Вычислительная сложность прямого аффинно-масштабирующего алгоритма  $A$  решения приведенной задачи (2) будет составлять  $O(n^{3,1})$ .

Результаты свидетельствуют, что алгоритм способен формально привести к точному решению за конечное число итераций. Однако в связи с экспоненциальным ростом времени работы алгоритма при устремлении к нулю заданной точности его применение для серии экспериментов с достаточно большим числом наблюдений ( $n \geq 50$ ) затруднительно.

## 2.2. Аффинно-масштабирующий алгоритм В

Решением двойственной задачи ЛП будем называть вектор  $\mathbf{u}$ , удовлетворяющий системе уравнений [16]

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

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

Положим

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

Алгоритм решения состоит в следующем. Пусть  $x_j^{(0)} > 0$ . Тогда  $x_j^{(k+1)} = x_j^{(k)} + \lambda_k s_j(\mathbf{x}^{(k)})$ , где  $\lambda_k = 1/\sqrt{\Phi(\mathbf{x}^{(k)})}$  при  $j = 1, \dots, m$ . Контроль вычислений на каждой итерации основывается на выполнении условия [31]

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

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

Решения примеров 1–3 при помощи описанного метода соответствуют точному вне зависимости от начальной точки  $\mathbf{x}^{(0)} > 0$ . Рассмотрим дополнительный пример.

*Пример 4.*  $F = \min(3x_1 + 2x_2 + x_3); \quad x_2 + x_3 \geq 4; \quad 2x_1 + x_2 + 2x_3 \geq 6;$   
 $2x_1 - x_2 + 2x_3 \geq 2; \quad x_i \geq 0, \quad i = 1, 2, 3.$  С точным решением  $F = 4$  в  $\mathbf{x} = (0; 0; 4)$  для прямой задачи ЛП и в  $\mathbf{y} = (1; 0; 0)$  для двойственной. Приведем результаты для первой и последней итераций алгоритма. На

первой итерации получим  $\mathbf{B}^{(1)} = \begin{pmatrix} 3 & 3 & 1 \\ 3 & 10 & 7 \\ 1 & 7 & 10 \end{pmatrix}$ ,  $\mathbf{d}^{(1)} = (3; 10; 6)$ , откуда

$\mathbf{u}^{(1)} = (-0,12; 1,19; -0,22)$  и  $\Phi(\mathbf{x}^{(1)}) = 3,78$ . Расчет для точности  $10^{-8}$  был

окончен за 15 итераций с результатами:  $\mathbf{B}^{(15)} = \begin{pmatrix} 5 & 8 & 8 \\ 8 & 16 & 16 \\ 8 & 16 & 20 \end{pmatrix}$ ,  $\mathbf{d}^{(15)} = (4; 8; 8)$ ,

откуда  $\mathbf{u}^{(15)} = (0; 0,5; 0)$  и  $\Phi(\mathbf{x}^{(15)}) = 4,5 \times 10^{-8}$  с решением  $F_{\max} \approx 3$ , что не соответствует точному, хотя условие (4) было выполнено на каждой итерации.

В результате проведения ряда вычислительных экспериментов, связанных с решением приведенной задачи (2), было выявлено, что решение при помощи алгоритма стремится к нулю. Это может являться следствием невыполнения условия в [32], согласно которому все ограничения-неравенства должны выполняться в строгой форме. Тем не менее определение зависимости среднего времени вычисления от размера выборки невозможно в связи с отсутствием сходимости метода.

*Утверждение 2. Вычислительная сложность одной итерации прямого аффинно-масштабирующего алгоритма В решения приведенной задачи (2) будет составлять  $O(n^3)$ , что с учетом числа итераций сопоставимо с алгоритмом А.*

Алгоритм градиентного спуска по узловым прямым описан в [26]. Он состоит в следующем. Пусть имеем  $m$ -мерное евклидово пространство  $\mathbb{R}^m$  со стандартным ортонормированным базисом  $\{\mathbf{e}_1, \dots, \mathbf{e}_m\}$ , где  $\mathbf{e}_k^T = \underbrace{(0, \dots, 0}_{k-1}, 1, \underbrace{0, \dots, 0}_{m-k})$ . Каждое невырожденное наблюдение  $(\mathbf{x}_i^T, y_i) = (1, x_{i2}, \dots, x_{im}, y_i)$ ,  $i = 1, \dots, n$ , формирует в  $\mathbb{R}^m$  гиперплоскость  $\Omega_i : y_i - \mathbf{a}^T \mathbf{x}_i = 0$  в ортогональной системе координат  $Oa_1 : a_m$ . Пересечение  $m$  независимых гиперплоскостей образует узловую точку

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

Пересечение  $(m - 1)$  независимых гиперплоскостей образует узловую прямую

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

Решение задачи (1) всегда находится в узловой точке. Алгоритм осуществляет спуск из произвольной начальной узловой точки по узловым прямым. Любая узловая точка применительно к задаче ЛП представляет собой ее базисное решение. Выигрыш по сравнению с симплекс-методом состоит в том, что переход по узловой прямой происходит более эффективно, в отличие от симплекс-метода здесь в преобразовании на каждом шаге участвуют не все данные, а только расположенные на соответствующей узловой прямой  $n$  точек. Кроме этого, на каждом шаге спуск осуществляется в точку с минимальным значением на всех пересекающих ее  $m$  узловых прямых.

С целью снижения вычислительных затрат при спуске анализируются не значения целевой функции, а ее производные по направлению узловой прямой. В модифицированном варианте алгоритма начальная узловая точка (начальное приближение) определяется на части выборки и исключены вычисления значений целевой функции в минимумах узловых прямых [27].

### 3. Обсуждение результатов

В результате сравнения алгоритмов внутренней точки с симплекс-методом можно утверждать, что в плане вычислительной сложности они им как ми-

нимум не уступают. Так, аффинно-масштабирующие алгоритмы А и В с учетом числа итераций имеют сложность  $O(n^{3,1})$ , в то время как решение прямой и двойственной задач ЛП при помощи симплекс-метода –  $O(n^{3,2}m^{0,2})$  и  $O(n^3m^{0,5})$  соответственно [15]. Однако при  $n \gg t$  они все еще значительно уступают модифицированному градиентному спуску, сложность которого равна  $O(n^{1,5}m^{1,8})$  [27].

Основным отличием градиентного спуска по узловым прямым от алгоритмов, основанных на решении задач ЛП при помощи симплекс-метода и алгоритмов внутренней точки, является оперирование только исходными данными вплоть до нахождения глобального минимума. Это гарантирует сходимость к точному решению вне зависимости от числа итераций и простоту реализации метода. Последние же на каждой последующей итерации используют информацию из предыдущей, содержащуюся в симплекс-таблице и векторах соответственно, что в результате накопления вычислительных погрешностей может привести к смещенной оценке. Тем не менее, учитывая этот факт при реализации, можно добиться пренебрежимо малого числа отклонений, однако это также увеличит вычислительные затраты.

Сравнение алгоритмов на основе оценки вычислительной сложности в нотации *Big O* необходимо, но недостаточно. Ее упрощенность приводит к игнорированию констант, множественных второстепенных слагаемых, а также потребления памяти алгоритмом. Поэтому могут возникать ситуации, когда два алгоритма с одинаковыми *Big O* будут иметь существенно отличающееся расчетное время вычислений или, наоборот, у алгоритмов с различными *Big O* окажется одинаковым расчетное время вычислений.

### 3.1. Сравнительный анализ алгоритмов на модельных данных

Для более полной оценки воспользуемся методом Монте-Карло и сравним время работы алгоритмов для 1000 экспериментов при  $t = 2, 3, \dots, 7$  и  $n = 50, 100, \dots, 500$ . Поскольку целью является исследование вычислительной эффективности алгоритмов, а не точности МНМ-оценивания регрессионных моделей, то в генерируемых выборках данных используем стандартное нормальное распределение случайных ошибок  $\varepsilon$ . Генерация произведена при помощи встроенных функций языка программирования С++ в среде Microsoft Visual Studio 2019. Вычислительные эксперименты будем проводить на ноутбуке Dell G5 5587 с 6-ядерным процессором i7-8750H с тактовой частотой до 4,1 ГГц. Точность для алгоритма А равна  $\delta = 10^{-5}$ .

В результате определения зависимости времени работы алгоритмов от числа наблюдений и параметров модели для аффинно-масштабирующего алгоритма А получим  $t_1 = 0,0001 \times n^{3,4}$  с коэффициентом детерминации  $R^2 = 0,96$ . Как было отмечено ранее, алгоритм В неприменим при решении приведенной задачи (2), в связи с чем исключим его из дальнейшего сравнения. Зависимости для решений прямой и двойственной задачи ЛП симплекс-методом, а также для модифицированного градиентного спуска будут составлять –  $t_2 =$

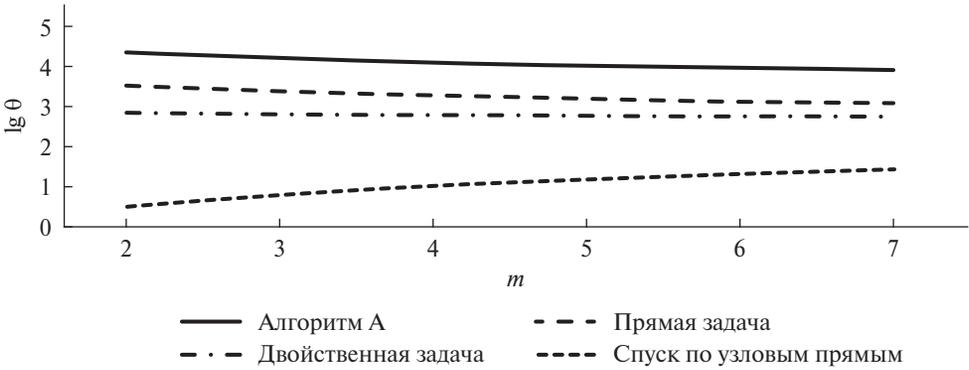


Рис. 1. Десятичные логарифмы отношений времени вычислений с помощью алгоритмов МНМ ко времени вычисления МНК при  $n = 300$ .

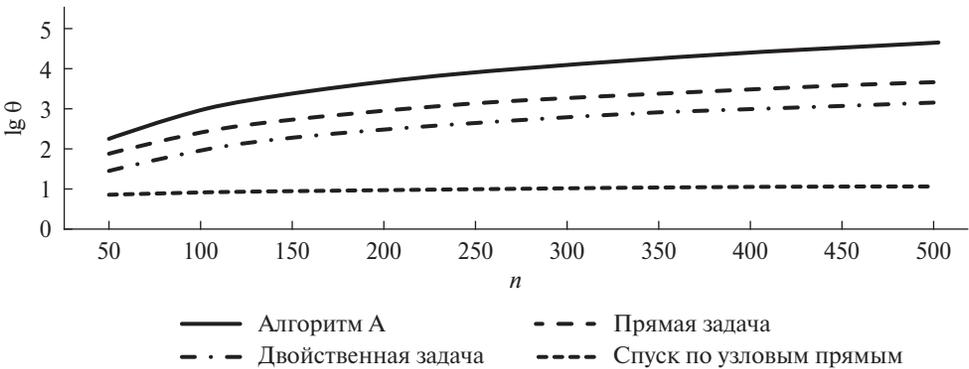


Рис. 2. Десятичные логарифмы отношений времени вычислений с помощью алгоритмов МНМ ко времени вычисления МНК при  $m = 4$ .

$= 0,0004 \times m^{0,001} n^{2,8}$ ,  $t_3 = 0,0001 \times m^{0,6} n^{2,7}$ ,  $t_4 = 0,00059 \times m^{2,51} n^{1,21}$  [27]. Для наглядности на рис. 1 и 2 приведем графики десятичных логарифмов отношений времени вычисления рассматриваемых алгоритмов ко времени расчета с помощью МНК.

В результате можно констатировать, что аффинно-масштабирующий алгоритм А на порядок уступает симплекс-методу и на несколько порядков – градиентному спуску.

### 3.2. Сравнительный анализ алгоритмов на практических примерах

Теперь рассмотрим три практических примера регрессионного моделирования на реальных данных. В качестве точного решения задачи (1) во всех примерах использовались результаты расчета с помощью полного перебора всех узловых точек [15]. Результаты расчетов с помощью симплекс-метода и алгоритма градиентного спуска по узловым прямым во всех трех примерах совпали с полным перебором. Таким образом, симплекс-метод и алгоритм градиентного спуска нашли точные решения  $\mathbf{a}^*$  задачи (1).

**Таблица 1.** Результаты расчетов параметров коэффициентов  $\mathbf{a}$  для примера 5

Алгоритм	$a_1$	$a_2$	$a_3$	$a_4$	$Q(\mathbf{a})$	$s$	$d$
Решение $\mathbf{a}^*$	-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%

**Таблица 2.** Результаты расчетов параметров коэффициентов  $\mathbf{a}$  для примера 6

Алгоритм	$a_1$	$a_2$	$a_3$	$a_4$	$Q(\mathbf{a})$	$s$	$d$
Решение $\mathbf{a}^*$	-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%

*Пример 5.* Определим параметры модели среднего экономического ущерба от пожаров в муниципальных образованиях (МО) Свердловской области за 2012 г. по данным, приведенным в [33]. Модель задается как  $\hat{Y} = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4$ , где  $\hat{Y}$  – средний прогноз ущерба от пожаров в текущем году, млн руб.;  $X_1 = 1$ ;  $X_2$  – количество зданий и сооружений на территории МО, тыс. шт.;  $X_3$  – общая протяженность автодорог на территории МО, км;  $X_4$  – годовые потери от пожаров в предыдущем году, млн руб.;  $\mathbf{a}$  – вектор искомых параметров модели;  $n = 58$ . Модель статистически значимая, коэффициент детерминации  $R^2 = 0,68$ . Результаты расчетов приведены в табл. 1.

*Пример 6.* Для оценки относительной производительности центральных процессоров (ЦП) в [34] приведены данные об их характеристиках и относительной производительности;  $n = 209$ . Машины представляли широкий спектр производительностей и производителей.

Прогнозирование относительной производительности ЦП выполнено по модели  $\hat{Y} = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4$ , где  $\hat{Y}$  – оценка относительной производительности ЦП;  $X_1 = 1$ ;  $X_2$  – размер основной памяти = (минимальный размер основной памяти + максимальный размер основной памяти)/2;  $X_3$  – размер кэш-памяти;  $X_4$  – пропускная способность канала = (минимальное количество каналов + максимальное количество каналов)/(2 × машинное время цикла);  $\mathbf{a}$  – вектор искомых параметров модели. Модель статистически значимая, коэффициент детерминации  $R^2 = 0,89$ . Результаты расчетов приведены в табл. 2.

**Таблица 3.** Результаты расчетов параметров коэффициентов **a** для примера 7

Алгоритм	$a_1$	$a_2$	$a_3$	$a_4$	$a_5$	$a_6$	$Q(\mathbf{a})$	$s$	$d$
Решение $\mathbf{a}^*$	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%

*Пример 7.* Рассмотрим задачу моделирования урожайности пшеницы в Киркуке по климатическим и социально-экономическим показателям по данным с 2020 по 2022 г. ( $n = 23$ ) [35]. Имеем регрессионную модель модели  $\hat{Y} = a_1X_1 + a_2X_2 + a_3X_3 + a_4X_4$ , где  $\hat{Y}$  – оценка урожайности пшеницы, т/га;  $X_1 = 1$ ;  $X_2$  – население Киркука, млн чел.;  $X_3$  – валовой внутренний продукт Ирака на душу населения в ценах 2023 г. (для учета инфляции), тыс. долл. США;  $X_4$  – нормализованный вегетационный индекс;  $X_5$  – поверхностное давление, кПа/10;  $X_6$  – скорость ветра на расстоянии не менее 10 метров, м/с;  $\mathbf{a}$  – вектор искомых параметров модели. Модель статистически значимая, коэффициент детерминации  $R^2 = 0,81$ . Результаты расчетов приведены в табл. 3.

В табл. 1–3 обозначено:  $s$  – среднее значение абсолютных относительных погрешностей вычисления коэффициентов  $\mathbf{a}$  (в %);  $d$  – отношение времени расчета алгоритмом А при точности  $\delta$  ко времени расчета при точности  $\delta_0 = = 10^{-3}$ .

Выводы по примерам 5–7:

- в рассмотренном диапазоне точности с уменьшением  $\delta$  время расчета растет в 2–2,5 раза;
- скорость сходимости алгоритма А к минимуму целевой функции  $Q(\mathbf{a})$  с увеличением размерности задачи и объема выборки данных снижается, причем более критичным является рост  $m$ .

#### 4. Заключение

На примере аффинно-масштабирующих алгоритмов В.И. Зоркальцева (алгоритм А) и И.И. Дикина (алгоритм В) была проанализирована эффективность методов внутренней точки для МНМ-оценивания регрессионных моделей.

Анализ алгоритмов внутренних точек А и В показал, что при решении задачи (1):

- их вычислительная сложность сопоставима с симплекс-методом, однако они проигрывают последнему по времени вычислений,

- они значительно (более чем на порядок) проигрывают модифицированному спуску по узловым прямым как по вычислительной сложности, так и по фактическому времени вычислений,
- скорость сходимости алгоритма  $A$  к минимуму целевой функции  $Q(\mathbf{a})$  с увеличением размерности задачи и объема выборки данных снижается, причем более критичным является рост размерности,
- рост точности увеличивает длительность работы алгоритма  $A$ , однако это менее критично по сравнению с размерностью задачи и объемом анализируемой выборки данных.

Результаты свидетельствуют, что алгоритм  $A$  без учета вычислительных погрешностей способен привести к точному решению за конечное число итераций. Однако зависимость времени работы от заданной точности, размерности задачи и объема выборки данных ограничивает область его применения для решения задач вида (1) величинами  $\delta \geq 10^{-8}$ ,  $m \leq 4$ ,  $n \leq 100$ .

Проигрыш алгоритма  $A$  симплекс-методу по времени при примерно одинаковой сложности вызван наличием у целевой функции в окрестности минимума множества почти параллельных очень малых граней, что усложняет и снижает эффективность использования барьеров.

В статье были рассмотрены только два алгоритма реализации методов внутренней точки. Но поскольку пока отсутствует информация о кардинальном (на порядок и более) фактическом повышении быстродействия современных алгоритмов, то можно утверждать, что спуск по узловым прямым для вычисления параметров линейных регрессионных моделей по экспериментальным данным более эффективен по сравнению с методами внутренней точки.

## ПРИЛОЖЕНИЕ

*Доказательство утверждения 1.* Вычислительная сложность умножения матриц  $(m \times n)$  на  $(n \times l)$  равна  $O(mnl)$ , а нахождения обратной методом Гаусса–Жордана для квадратной матрицы размерности  $(n \times n)$  равна  $O(n^3)$ . Пусть  $n_1 = 2n$ ;  $n_2 = 3n + 2m$ , тогда размеры матриц для приведенной задачи (2) будут:  $\mathbf{A} - (n_1 \times n_2)$ ;  $\mathbf{x} - (n_2 \times 1)$ ;  $\mathbf{b} - (n_1 \times 1)$ ;  $\mathbf{c} - (n_2 \times 1)$ . Учтем, что при  $n \gg m$  в нотации *Big O* вес числа операций  $n_1$  и  $n_2$  будет сопоставим. Определим сложность каждого шага алгоритма для решения приведенной к ЗЛП задачи (1):

*Шаг 1.*  $\mathbf{Ax}^{(k)}$  является умножением матрицы размером  $(n_1 \times n_2)$  на  $(n_2 \times 1)$  и будет вычисляться за  $O(n_1 n_2)$  операций. Вместе с операцией вычитания полная сложность шага будет составлять  $O(n_1 + n_1 n_2)$  или  $P_1 = O(n_1 n_2)$ .

*Шаг 2.* Производится возведение в степень  $p$  вектора  $\mathbf{x}$  и формирование из него диагональной матрицы. Так как возведение в  $p = 2$  равноценно умножению числа на само себя, его сложность равна  $P_2 = O(n_2)$ , что и будет составлять сложность шага в силу того, что диагональная матрица задается путем инициализации из полученного вектора.

*Шаг 3.* Разберем его поэтапно:

1) произведение матриц  $\mathbf{A}\mathbf{D}_k\mathbf{A}^T$  будет вычислено за  $(n_1n_2^2 + n_1^2n_2)$  операций;

2) обратная от найденной матрицы будет определена за  $(n_1^3)$  операций;

3) выражение  $(\mathbf{r}^{(k)} + \mathbf{A}\mathbf{D}_k\mathbf{c})$  за  $(n_1n_2^2 + n_1n_2 + n_1)$  операций;

4) умножение полученных матриц будет выполнено за  $(n_1^2)$  операций.

Общая вычислительная сложность будет равна  $O(n_1^3 + 2n_1n_2^2 + n_1^2n_2 + n_1^2 + n_1n_2 + n_1)$  или  $P_3 = O(n_1^3)$  для  $n \gg m$ .

*Шаг 4.* Вектор  $\mathbf{g}(\mathbf{u}^{(k)})$  находится за  $(n_1n_2 + n_2)$  операций. Тогда общая сложность будет равна  $O(n_2^2 + n_1n_2 + n_2)$  или  $P_4 = O(n_2^2)$ .

*Шаг 5.* Итеративный переход осуществляется за  $P_5 = O(n_2)$ .

Шаги 2–5 находятся в цикле вплоть до достижения точки останова. Таким образом, общая вычислительная сложность алгоритма будет составлять:

$$\begin{aligned} P &= P_1 + (P_2 + P_3 + P_4 + P_5) \times \{\text{число итераций}\} = \\ &= n_1n_2 + (n_2 + n_1^3 + n_2^2 + n_2) \times \{\text{число итераций}\}, \end{aligned}$$

где число итераций является константой, зависящей от заданной точности алгоритма.

Методом статистических испытаний Монте-Карло установлено, что оптимальной заданной точностью решения при стандартном нормальном распределении случайных ошибок  $\delta$  в анализируемой выборке является  $10^{-5}$ , в связи с низкой точностью при  $10^{-3}$  и длительным временем вычислений при  $10^{-8}$ . Таким образом, алгоритм останавливается при достижении максимального по модулю значения вектора  $\mathbf{s}$ , меньшего чем  $10^{-5}$ , что позволяет сохранить приемлемый уровень точности решения вне зависимости от размера анализируемой выборки.

Для 100 итераций при  $m = 2, 3, \dots, 7$  и  $n = 50, 100, \dots, 500$  в среднем алгоритм будет находить решение за 5 итераций с отклонением от точного решения на 14,3%. Для учета возможного влияния числа итераций на вычислительную сложность алгоритма вне зависимости от заданной точности решения (при  $n$ , значительно превосходящем число итераций) увеличим степень при  $n$

$$O(P) = O(n_1^3 \times \{\text{число итераций}\}) \leq O(n^{3,1}).$$

Утверждение 1 доказано.

*Доказательство утверждения 2.* Определим вычислительную сложность одной итерации при условии, что решается приведенная задача (2) – для определения двойственных оценок вычисляется матрица  $\mathbf{B}$  за  $O(4n_1^2n_2)$  и вектор  $\mathbf{d}$  за  $O(4n_1n_2)$  операций соответственно.

Таким образом, вычисление вектора решений  $\mathbf{u}$  происходит за  $O(n_1^3 + 4n_1^2n_2 + n_1^2 + 4n_1n_2)$  или за  $O(n_1^3)$  операций. Вектор  $\mathbf{s}(\mathbf{x})$  будет найден за  $O(2n_2(2n_1 + 1))$  или  $O(n_1n_2)$ , в то время как  $\Phi(\mathbf{x})$  – за  $O(3n_2(2n_1 + 1))$  или  $O(n_1n_2)$  операций.

Итоговая вычислительная сложность одной итерации будет составлять  $O(n_1^3)$  или  $O(n^3)$  операций. С учетом возможного числа итераций в зависимости от заданной точности решения метод действительно можно считать сопоставимым с алгоритмом А.

Утверждение 2 доказано.

## СПИСОК ЛИТЕРАТУРЫ

1. *Greene W.H.* Econometric Analysis: 8th ed. New York. Pearson. 2020.
2. *Айвазян С.А., Енюков И.С., Мешалкин Л.Д.* Прикладная статистика: Исследование зависимостей. М.: Финансы и статистика, 1985. С. 488.
3. Робастные методы статистического анализа навигационной информации: Обзор. Авторы-сост. Н.В. Бабкин, А.А. Мусаев, А.В. Макшанов; Под ред. И.Б. Челпанова. Л.: ЦНИИ «Румб», 1985. 206 с.
4. *Орлов А.И.* О требованиях к статистическим методам анализа данных (обобщающая статья) // Заводская лаборатория. Диагностика материалов. 2023. Т. 89. № 11. С. 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., Pal A.* Statistical Characterization of Random Errors Present in Synchrophasor Measurements // IEEE Power & Energy Society General Meeting (PESGM), Washington, DC, USA. 2021. P. 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 Ecology and Evolution. 2022. V. 13. No. 10. P. 2092–2105. <https://doi.org/10.1111/2041-210X.13971>
7. *Колобов А.Б.* Вибродиагностика: теория и практика. М.: Инфра-Инженерия, 2019. 252 с.
8. *Dodge Y.* The Concise Encyclopedia of Statistics. Springer. 2008. P. 616. <https://doi.org/10.1007/978-0-387-32833-1>
9. *Акимов П.А., Матасов А.И.* Итерационный алгоритм для  $l_1$ -аппроксимации в динамических задачах оценивания // АиТ. 2015. № 5. С. 7–26.
10. *Граничнин О.Н., Поляк Б.Т.* Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах. М.: Наука, 2003. 291 с.
11. *Назин А.В., Немировский А.С., Цыбаков А.Б., Юдицкий А.Б.* Алгоритмы робастной стохастической оптимизации на основе метода зеркального спуска // АиТ. 2019 № 10. С. 78–99. <https://doi.org/10.1134/S000523101909006X>
12. *Guo L., Ljung L.* Performance analysis of general tracking algorithms // IEEE Transactions on Automatic Control. 1995. V. 40. No. 8. P. 138–1402. <https://doi.org/10.1109/9.402230>
13. *Campi M.C., Weyer E.* Guaranteed non-asymptotic confidence regions in system identification // Automatica. 2005. V. 41. No. 10. P. 1751–1764. <https://doi.org/10.1016/j.automatica.2005.05.005>
14. *Мудров В.И., Кушко В.Л.* Методы обработки измерений: Квазиравдоподобные оценки. М.: URSS, 2022. С. 304.
15. *Голованов О.А., Тырсин А.Н.* Спуск по узловым прямым и симплекс-алгоритм – два варианта регрессионного анализа на основе метода наименьших модулей // Заводская лаборатория. Диагностика материалов. 2024. Т. 90. № 5. С. 79–87. <https://doi.org/10.26896/1028-6861-2024-90-5-79-87>

16. *Дикин И.И.* Итеративное решение задач линейного и квадратичного программирования // Доклады АН СССР. 1967. Т. 174. № 4. С. 747–748.
17. *Karmarkar N.* A New Polynomial-Time Algorithm for Linear Programming // *Combinatorica*. 1984. V. 4. No. 4. P. 373–395. <https://doi.org/10.1145/800057.808695>
18. *Bayer D.A., Lagarias J.C.* The Nonlinear Geometry of Linear Programming I: Affine and Projective Scaling Trajectories // *Transactions of the American Mathematical Society*. 1989. V. 314. P. 499–526. <https://doi.org/10.1090/S0002-9947-1989-1005525-6>
19. *Воронцова Е.А., Хильдебрандт Р.Ф., Гасников А.В., Стонякин Ф.С.* Выпуклая оптимизация. М.: МФТИ, 2021. С. 364.
20. *Зоркальцев В.И.* Метод внутренних точек: история и перспективы // Журн. вычисл. мат. и мат. физики. 2019. Т. 59. № 10. С. 1649–1665. <https://doi.org/10.1134/S0044466919100181>
21. *Nesterov Yu., Nemirovski A.* Interior-Point Polynomial Algorithms in Convex Programming // *SIAM Studies in Applied Mathematics*. Philadelphia: SIAM. 1994. V. 13. P. 414.
22. *Lee Y.T., Yue M.-C.* Universal Barrier is  $n$ -Self-Concordant // E-print. 2018. <https://doi.org/10.48550/arXiv.1809.03011>
23. *Bubeck S., Eldan R.* The Entropic Barrier: Exponential Families, Log-Concave Geometry, and Self-Concordance // *Mathematics of Operations Research*. 2019. V. 44. No. 1. P. 264–276. <https://doi.org/10.1287/moor.2017.0923>
24. *Панюков А.В., Мезал Я.А.* Параметрическая идентификация квазилинейного разностного уравнения // Вест. Южно-Урал. гос. ун-та. Серия: Математика. Механика. Физика. 2019. Т. 11. № 4. С. 32–38. <https://doi.org/10.14529/mmph190404>
25. *Barrodale I., Roberts F.D.K.* An improved algorithm for discrete  $L_1$  linear approximation // *SIAM J. Numer. Anal.* 1973. V. 10. P. 839–848.
26. *Тырсин А.Н.* Алгоритмы спуска по узловым прямым в задаче оценивания регрессионных уравнений методом наименьших модулей // Заводская лаборатория. Диагностика материалов. 2021. Т. 87. № 5. С. 68–75. <https://doi.org/10.26896/1028-6861-2021-87-5-68-75>
27. *Голованов О.А., Тырсин А.Н.* Модификация метода наименьших модулей на основе градиентного спуска по узловым прямым // Математические методы в технологиях и технике. 2023. № 11. С. 43–46. [https://doi.org/10.52348/2712-8873\\_MMTT\\_2023\\_11\\_43](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 // *Communications in Statistics, Simulation and Computation*. 1981. V. B10. No. 5. P. 479–491.
29. *Li Y., Arce G.R.* A Maximum Likelihood Approach to Least Absolute Deviation Regression // *EURASIP J. Advan. Signal Proc.* 2004. V. 12. P. 1762–1769. <https://doi.org/10.1155/S1110865704401139>
30. *Wei Xue, Wensheng Zhang, Gaohang Yu.* Least absolute deviations learning of multiple tasks // *J. Indust. Management Optim.* 2018. No. 14(2). P. 719–729. <https://doi.org/10.3934/jimo.2017071>
31. *Зоркальцев В.И., Дикин И.И.* Итеративное решение задач математического программирования (алгоритмы метода внутренних точек). Новосибирск: Наука, 1980. 144 с.

32. *Филатов А.Ю.* Развитие алгоритмов внутренних точек и их приложение к системам неравенств // Дисс. ... канд. физ.-мат. наук, 05.13.18. Иркутск. 2001. С. 123.
33. *Тужиков Е.Н.* Методика оценки эффективности деятельности органов местного самоуправления по обеспечению первичных мер пожарной безопасности (на примере Свердловской области) // Дисс. ... канд. техн. наук, 05.13.10. Екатеринбург. 2014. С. 186.
34. *Ein-Dor P., Feldmesser J.* Attributes of the Performance of Central Processing Units: a Relative Performance Prediction Model // Communications of the ACM. 1987. V. 30. No. 4. P. 308–317.
35. *Тырсин А.Н., Хасан Х.А.М.* Анализ факторов, влияющих на урожайность зерновых культур в Ираке (на примере Киркука) // Математические методы в технике и технологиях. 2024. № 8. С. 81–87.

*Статья представлена к публикации членом редколлегии А.А. Бобцовым.*

Поступила в редакцию 14.03.2024

После доработки 28.11.2024

Принята к публикации 02.12.2024