2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

doi 10.26089/NumMet.v??r??? УДК 519.852

О новом методе линейного программирования

Н. А. Ольховский
Южно-Уральский государственный университет (национальный исследовательский университет), Челябинск, Российская Федерация
ORCID: 0009-0008-9078-4799, e-mail: olkhovskiina@susu.ru

Л. Б. Соколинский
Южно-Уральский государственный университет (национальный исследовательский университет), Челябинск, Российская Федерация
ORCID: 0000-0001-9997-3918, e-mail: leonid.sokolinsky@susu.ru

Аннотация: В статье представлен новый итерационный метод линейного программирования, получивший название “метод поверхностного движения”. Данный метод строит на поверхности многогранника, ограничивающего допустимую область задачи линейного программирования, путь от начальной граничной точки до точки, в которой достигается оптимальное значение целевой функции. Вектор движения строится в направлении максимального увеличения/уменьшения значения целевой функции. Представлено формальное описание алгоритма, реализующего метод поверхностного движения. Доказана теорема сходимости. Описанный метод предполагает использование глубокой нейронной сети прямого распространения для определения направления движения по граням допустимого многогранника. Для этого строится многомерный локальный образ задачи линейного программирования в точке текущего приближения, который подается на вход нейронной сети. Множество размеченных прецедентов, необходимое для обучения нейронной сети может быть получено с помощью апекс-метода.

Ключевые слова: линейное программирование, метод поверхностного движения, итерационный метод, теорема сходимости, глубокая нейронная сеть.

Благодарности: Исследование выполнено при финансовой поддержке РНФ (проект № 23-21-00356).

Для цитирования: Ольховский Н.А., Соколинский Л.Б. О новом методе линейного программирования // Вычислительные методы и программирование. 2023. **24**, № ?. 1–26. doi 10.26089/NumMet.v??r???.

A new method of linear programming

N. A. Olkhovsky
South Ural State University (National Research University), Chelyabinsk, Russia
ORCID: 0009-0008-9078-4799, e-mail: olkhovskiina@susu.ru

L. B. Sokolinsky
South Ural State University (National Research University), Chelyabinsk, Russia
ORCID: 0000-0001-9997-3918, e-mail: leonid.sokolinsky@susu.ru

Abstract: The article presents a new iterative method of linear programming, called the surface movement method. This method builds a path from the initial boundary point to the point at which the optimal value of the objective function is achieved on the surface of a polytope that restricts the feasible region of linear programming problem. The movement vector defines the direction of

2021, **??** (?), 1–26. doi 10.26089/NumMet.v??r???

the maximum increase/decrease in the value of the objective function. A formal description of the algorithm implementing the surface movement method is ed. The convergence theorem is proved. The described method involves the use of a feed forward deep neural network to determine the direction of movement along the edges of a feasible polytope. To do this, a multidimensional local image of the linear programming problem is constructed at the point of the current approximation, which is fed to the input of the neural network. The set of labeled precedents necessary for training a neural network can be obtained using the apex method.

Keywords: linear programming, surface motion method, iterative method, convergence theorem, deep neural network.

Acknowledgements: The research was supported by RSF (project No. 23-21-00356).

For citation: N.A. Olkhovsky and L.B. Sokolinsky, “A new method of linear programming using neural networks”, Numerical Methods and Programming. **24** (?), 1–26 (2023). doi 10.26089/NumMet.v??r???.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

1. Введение. Быстрое развитие технологий обработки и хранения больших данных [1] привело к возникновению оптимизационных математических моделей в виде задач линейного программирования (ЛП) большой размерности [2]. Особый интерес представляют нестационарные задачи ЛП, связанные с оптимизацией нестационарных процессов [3]. В нестационарной задаче ЛП ограничения и целевая функция могут меняться динамически в ходе ее решения [4]. Следующие оптимизационные задачи могут быть сведены к нестационарным задачам ЛП: выбор оптимальных стратегий в роботрейдинге [5, 6], оптимальное управление летательными аппаратами [7], оптимизация технологических процессов [8, 9, 10], логистические и транспортные задачи [11, 12, 13], планирование и управление производством продукции [14]. Отдельно можно упомянуть оптимизационные задачи, которые должны решаться в режиме реального времени [15]. В качестве примеров можно привести управление химическим производством, управление системой многоточечного впрыска топлива в ДВС, управление сотовыми сетями, автопилотирование, системы самонаведения ракет.

Простейший подход к решению нестационарных задач оптимизации заключается в том, что всякое изменение исходных данных воспринимается как отдельная новая задача [3]. Такой подход может быть приемлемым, когда изменения происходят относительно медленно, а оптимизационная задача решается относительно быстро. Однако для больших нестационарных оптимизационных задач решение, получаемое таким способом, оказывается далеким от оптимального в силу изменения исходных данных в процессе вычислений. В этом случае необходимо использовать алгоритмы, динамически корректирующие вычислительный процесс в соответствии с изменяющимися исходными данными. Тем самым, вычисления с измененными данными начинаются не с нуля, а используют информацию, полученную в прошлом. Такой подход применим для решения задач реального времени при условии, что алгоритм достаточно быстро отслеживает движение точки оптимума. Для больших задач ЛП последнее требование делает актуальной разработку масштабируемых методов и параллельных алгоритмов линейного программирования.

До настоящего времени одним из самых популярных способов решения задач линейного программирования является семейство алгоритмов, разработанных на основе симплекс-метода [16]. Симплекс-метод способен решать большие задачи ЛП, эффективно используя различные виды гиперразреженности [17]. Однако симплекс методу присущ ряд фундаментальных недостатков. Во-первых, на некоторых задачах симплекс-методу приходится обходить все вершины симплекса, что соответствует экспоненциальной временной сложности [18]. Во-вторых, при решении симплекс-методом очень больших задач ЛП, размерность которых превышает 50 000, часто наблюдается потеря точности [19], которую не удается компенсировать применением даже таких мощных алгоритмов, как аффинное масштабирование или итеративное уточнение [20]. В-третьих, информационная структура алгоритмов, основанных на симплекс-методе, имеет ограниченный ресурс параллелизма, что делает невозможным их эффективное распараллеливание на больших вычислительных системах с распределенной памятью [21, 22]. Все перечисленное затрудняет использование симплекс-метода для решения нестационарных задач ЛП в режиме реального времени.

Другим популярным подходом к решению больших задач ЛП является класс алгоритмов, основанных на методе внутренних точек [23], предложенном Дикиным [24]. Эти алгоритмы способны решать задачи ЛП с миллионами переменных и ограничений [25]. Достоинством метода внутренних точек является то, что он самокорректируется и способен обеспечить высокую точность вычислений. Основными недостатками метода внутренних точек являются следующие. Во-первых, значимый подкласс алгоритмов, основанных на методе внутренних точек, требует в качестве начального приближения некоторую внутреннюю точку допустимой области задачи ЛП. Нахождение такой точки можно свести к решению дополнительной задачи ЛП [26]. Другим способом нахождения внутренних точек является использование фейеровских приближений [27]. Во-вторых, метод внутренних точек плохо масштабируется в больших вычислительных системах кластерного типа. Известны некоторые частные случае, когда удается выполнить эффективное распараллеливание метода внутренних точек (см., например, [28]), но в общем случае построит эффективную параллельную реализацию этого метода для кластерных вычислительных систем не удается. В-третьих, итерационный характер метода внутренних точек не позволяет заранее предсказать время вычислений для конкретной задачи ЛП. Указанные недостатки затрудняют применения метода внутренних точек для решения больших задач ЛП в режиме реального времени.

Новым перспективным подходом к решению оптимизационных задач, вызывающим большой интерес, являются искусственные нейронные сети [29], представляющие собой мощный универсальный инструмент, применимый практически во всех проблемных областях. Одним из первых применений искусственных нейронных сетей к решению задач ЛП была работа Хопфилда и Танка [30]. Нейронная сеть

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Хопфилда-Танка состоит из двух полносвязных слоев и является рекуррентной. Число нейронов первого слоя совпадает с числом переменных задачи ЛП. Число нейронов во втором слое равно числу ограничений. Веса и смещения нейронной сети полностью определяются параметрами задачи линейного программирования. Выходной сигнал циклически подается на вход нейронной сети. Нейронная сеть работает до достижения состояния равновесия, когда выход становится равным входу. Это состояние соответствует минимум специальной энергетической функции и является решением задачи ЛП. Подход Хопфилда-Танка был развит в большом количестве работ (см., например, [31, 32, 33, 34, 35]). Главным недостатком этого подхода является то, что невозможно предсказать количество циклов работы нейронной сети, необходимое для достижения состояния равновесия. Это делает невозможным использование таких рекуррентных сетей для решения больших задач ЛП в режиме реального времени. Для этой цели более перспективными представляются глубокие нейронные сети прямого распространения. Архитектура и параметры таких сетей, как правило, не зависят от входных данных задачи. Решение получается за один проход с фиксированным временем работы сети, что обеспечивает возможность их применения для решения задач в режиме реального времени. Особый интерес представляют сверточные нейронные сети [36], ориентированные на распознавание и классификацию образов. В недавней работе [37] был предложен оригинальный метод построения образов многомерных задач ЛП, открывающий возможность использовать нейронные сети прямого распространения, включая сверточные, для их решения. Необходимо отметить, что глубокие нейронные сети требуют обучения на большом количестве размеченных прецедентов, которое может быть эффективно выполнено на графических процессорах [38]. В статье [39] предложен итерационный апекс-метод решения задач ЛП, позволяющий построить на поверхности допустимого многогранника путь в направлении максимального увеличения/уменьшения значения целевой функции, приводящий к точке оптимума. Апекс-метод принадлежит к классу проекционных методов, для которых характерна низкая линейная скорость сходимости, что делает их неприемлемыми для режима реального времени. Однако апекс-метод позволяет построить размеченное множество прецедентов, для обучения нейронных сетей прямого распространения.

В данной статье представлен новый итерационный метод метод решения задач ЛП, получивший название “метод поверхностного движения”. Указанный метод ориентирован на использование искусственных нейронных сетей прямого распространения, в том числе сверточных нейронных сетей. Статья организована следующим образом. В разделе 2 представлен теоретический базис, на котором основан метод поверхностного движения. Раздел 3 посвящен описанию метода поверхностного движения и доказательству теоремы сходимости. В разделе 4 обсуждаются сильные и слабые стороны предложенного метода, а также раскрываются пути его практической реализации на основе синтеза суперкомпьютерных и нейросетевых технологий. В разделе 5 суммируются полученные результаты и приводятся направления дальнейших исследований. Сводка обозначений, используемых в статье, приведена в разделе 6.

2. Теоретический базис. В этом разделе описывается теоретический фундамент, на котором базируется метод поверхностного движения.

Сформулируем задачу ЛП в следующем виде:

$$\bar{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \langle \boldsymbol{c}, \boldsymbol{x} \rangle \,|\, A\boldsymbol{x} \leqslant \boldsymbol{b} \right\}, \tag{1}$$

где $\boldsymbol{c} \in \mathbb{R}^n$, $\boldsymbol{b} \in \mathbb{R}^m$, $A \in \mathbb{R}^{m\times n}$, $m > 1$, $\boldsymbol{c} \neq \boldsymbol{0}$. Здесь $\langle \cdot, \cdot \rangle$ обозначает скалярное произведение двух векторов. Мы предполагаем, что ограничение $\boldsymbol{x} \geqslant \boldsymbol{0}$ также включено в матричное неравенство $A\boldsymbol{x} \leqslant \boldsymbol{b}$ в форме

$$-\boldsymbol{x} \leqslant \boldsymbol{0}.$$

Обозначим через $\mathcal{P}$ множество индексов, нумерующих строки матрицы A:

$$\mathcal{P} = \{1, \cdots, m\}.$$

Линейная целевая функция задачи (1) имеет вид

$$f(\boldsymbol{x}) = \langle \boldsymbol{c}, \boldsymbol{x} \rangle.$$

Вектор $\boldsymbol{c}$ в данном случае является градиентом целевой функции $f(\boldsymbol{x})$.

Пусть $\boldsymbol{a}_i \in \mathbb{R}^n$ обозначает вектор, представляющий i-тую строку матрицы A. Мы предполагаем, что $\boldsymbol{a}_i \neq \boldsymbol{0}$ для всех $i \in \mathcal{P}$. Обозначим через $\hat{H}_i$ замкнутое полупространство, определяемое неравенством

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

$\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \leqslant b_i$, а через H_i — ограничивающую его гиперплоскость:

$$\hat{H}_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \leqslant b_i\}; \tag{2}$$

$$H_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle = b_i\}. \tag{3}$$

Определим допустимый многогранник

$$M = \bigcap_{i \in \mathcal{P}} \hat{H}_i, \tag{4}$$

представляющий множество допустимых точек задачи ЛП (1). Заметим, что M в этом случае будет замкнутым выпуклым множеством. Мы будем предполагать, что множество M является ограниченным и $M \neq \emptyset$, то есть задача ЛП (1) имеет решение. Обозначим через $\Gamma(M)$ множество граничных точек многогранника M[1].

Утверждение 1. Пусть M — допустимый многогранник задачи ЛП (1), определяемый формулой (4). Тогда для любой точки $\boldsymbol{u} \in \Gamma(M)$ существует $\epsilon > 0$ такой, что для любой граничной точки $\boldsymbol{w}$, принадлежащей ϵ-окрестности $V_\epsilon(\boldsymbol{u})$ точки $\boldsymbol{u}$, найдется $i' \in \mathcal{P}$, для которого справедливо $\boldsymbol{u}, \boldsymbol{w} \in H_{i'}$:

$$\forall \boldsymbol{u} \in \Gamma(M)\ \exists \epsilon > 0 : \forall \boldsymbol{w} \in V_\epsilon(\boldsymbol{u}) \cap \Gamma(M)\ \exists i' \in \mathcal{P} : \boldsymbol{u}, \boldsymbol{w} \in H_{i'}.$$

Доказательство. Зафиксируем произвольную точку $\boldsymbol{u} \in \Gamma(M)$. Обозначим

$$\mathcal{P}_{\boldsymbol{u}} = \{ i \in \mathcal{P} |\, \boldsymbol{u} \in H_i \}. \tag{5}$$

Другими словами, $\mathcal{P}_{\boldsymbol{u}}$ — множество индексов всех гиперплоскостей H_i, которым принадлежит точка $\boldsymbol{u}$. Положим

$$\mathcal{P}_{\setminus \boldsymbol{u}} = \mathcal{P} \setminus \mathcal{P}_{\boldsymbol{u}},$$

то есть $\mathcal{P}_{\setminus \boldsymbol{u}}$ — множество индексов всех гиперплоскостей H_i, которым точка $\boldsymbol{u}$ не принадлежит. Определим

$$\delta = \min \left\{ dist(\boldsymbol{u}, H_i) |\, i \in \mathcal{P}_{\setminus \boldsymbol{u}} \right\},$$

где $dist(\boldsymbol{u}, H_i)$ обозначает евклидово расстояние от точки $\boldsymbol{u}$ до гиперплоскости H_i[2]. По определению

$$\delta > 0.$$

Возьмем ϵ, удовлетворяющий условию

$$0 < \epsilon < \delta.$$

Тогда для любого $\boldsymbol{w} \in V_\epsilon(\boldsymbol{u}) \cap \Gamma(M)$ имеем

$$\forall i \in \mathcal{P}_{\setminus \boldsymbol{u}} : \boldsymbol{w} \notin H_i.$$

Поскольку $\boldsymbol{w}$ — граничная точка, то отсюда следует, что найдется $i' \in \mathcal{P}_{\boldsymbol{u}}$ такой, что

$$\boldsymbol{w} \in H_{i'}.$$

Заметим, что в силу (5) также имеем

$$\boldsymbol{u} \in H_{i'}.$$

Утверждение доказано. □

Определение 1. Целевой проекцией точки $\boldsymbol{z} \in \mathbb{R}^n$ на гиперплоскость H_i называется точка $\boldsymbol{\gamma}_i(\boldsymbol{z}) \in \mathbb{R}^n \cup \{\infty\}$, определяемая формулой

$$\boldsymbol{\gamma}_i(\boldsymbol{z}) = \begin{cases} L(\boldsymbol{z}) \cap H_i, & \text{если } \langle \boldsymbol{a}_i, \boldsymbol{c} \rangle \neq 0; \\ \infty, & \text{если } \langle \boldsymbol{a}_i, \boldsymbol{c} \rangle = 0, \end{cases} \tag{6}$$

где $L(\boldsymbol{z})$ — прямая, проходящую через точку $\boldsymbol{z}$ параллельно вектору $\boldsymbol{c}$:

$$L(\boldsymbol{z}) = \{ \boldsymbol{y} \in \mathbb{R}^n |\, \boldsymbol{y} = \boldsymbol{z} + \lambda \boldsymbol{c}, \lambda \in \mathbb{R} \}. \tag{7}$$

[1] Под граничной точкой множества $M \subset \mathbb{R}^n$ понимается точка в $\mathbb{R}^n$, для которой любая открытая ее окрестность в $\mathbb{R}^n$ имеет непустое пересечение как с множеством M, так и с его дополнением.

[2] В данном случае $dist(\boldsymbol{u}, H_i) = \frac{\langle \boldsymbol{a}_i, \boldsymbol{u} \rangle - b_i}{\|\boldsymbol{a}_i\|}$.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

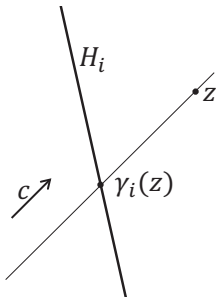

Рис. 1. Целевая проекция $\boldsymbol{\gamma}_i(\boldsymbol{z})$ точки $\boldsymbol{z}$ на гиперплоскость H_i

Fig. 1. Objective projection $\boldsymbol{\gamma}_i(\boldsymbol{z})$ of point $\boldsymbol{z}$ onto hyperplane H_i

Другими словами, если вектор $\boldsymbol{c}$ не параллелен гиперплоскости H_i, то целевой проекцией точки $\boldsymbol{z}$ на гиперплоскость H_i является точка пересечения этой гиперплоскости с прямой, проходящей через точку $\boldsymbol{z}$ параллельно вектору $\boldsymbol{c}$ (см. рис. 1). В случае, когда вектор $\boldsymbol{c}$ параллелен гиперплоскости H_i, целевая проекция полагается равной бесконечно удаленной точке ∞.

Следующее утверждение предоставляет формулу для вычисления целевой проекции $\boldsymbol{\gamma}_i(\boldsymbol{z})$ точки $\boldsymbol{z}$ на гиперплоскость H_i.

Утверждение 2. Пусть $\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle \neq 0$. Тогда

$$\boldsymbol{\gamma}_i(\boldsymbol{z}) = \boldsymbol{z} - \frac{\langle \boldsymbol{a}_i, \boldsymbol{z} \rangle - b_i}{\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle} \boldsymbol{c}. \tag{8}$$

Доказательство. В соответствии с (6) и (7) имеем

$$\boldsymbol{\gamma}_i(\boldsymbol{z}) = \boldsymbol{z} + \lambda \boldsymbol{c} \tag{9}$$

при некотором $\lambda \in \mathbb{R}$. С другой стороны, в соответствии с (3) имеем

$$\langle \boldsymbol{a}_i, \boldsymbol{\gamma}_i(\boldsymbol{z}) \rangle = b_i. \tag{10}$$

Подставим правую часть формулы (9) в формулу (10) вместо $\boldsymbol{\gamma}_i(z)$:

$$\langle \boldsymbol{a}_i, \boldsymbol{z} + \lambda \boldsymbol{c} \rangle = b_i.$$

Отсюда

$$\lambda = -\frac{\langle \boldsymbol{a}_i, \boldsymbol{z} \rangle - b_i}{\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle}. \tag{11}$$

Подставив правую часть формулы (11) вместо λ в формулу (9), получаем

$$\boldsymbol{\gamma}_i(\boldsymbol{z}) = \boldsymbol{z} - \frac{\langle \boldsymbol{a}_i, \boldsymbol{z} \rangle - b_i}{\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle} \boldsymbol{c}.$$

Утверждение доказано. □

Определение 2. Целевым смещением точки $\boldsymbol{z} \in \mathbb{R}^n$ относительно гиперплоскости H_i называется скалярная величина $\beta_i(\boldsymbol{z})$, вычисляемая по формуле

$$\beta_i(\boldsymbol{z}) = -\frac{\langle \boldsymbol{a}_i, \boldsymbol{z} \rangle - b_i}{\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle} \|\boldsymbol{c}\|. \tag{12}$$

Далее мы для краткости будем использовать термин “смещение”, подразумевая по этим “целевое смещение”. Обозначим

$$\boldsymbol{e}_{\boldsymbol{c}} = \frac{\boldsymbol{c}}{\|\boldsymbol{c}\|}. \tag{13}$$

Тогда формулу (8) можно переписать в виде

$$\boldsymbol{\gamma}_i(\boldsymbol{z}) = \boldsymbol{z} + \beta_i(\boldsymbol{z}) \boldsymbol{e}_{\boldsymbol{c}}, \tag{14}$$

что равносильно

$$\beta_i(\boldsymbol{z}) \boldsymbol{e}_{\boldsymbol{c}} = \boldsymbol{\gamma}_i(\boldsymbol{z}) - \boldsymbol{z}.$$

С учетом (13) отсюда следует

$$|\beta_i(\boldsymbol{z})| = \|\boldsymbol{\gamma}_i(\boldsymbol{z}) - \boldsymbol{z}\|. \tag{15}$$

Таким образом, $|\beta_i(\boldsymbol{z})|$ является расстоянием от точки $\boldsymbol{z}$ до ее целевой проекции на гиперплоскость H_i.

Определение 3. Целевой гиперплоскостью $H_c(\boldsymbol{z})$, проходящей через точку $\boldsymbol{z}$, будем называть гиперплоскость, задаваемую формулой

$$H_c(\boldsymbol{z}) = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle\}. \tag{16}$$

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Справедливо следующее утверждение.

Утверждение 3. Зафиксируем произвольную точку $\boldsymbol{z} \in \mathbb{R}^n$. Тогда для любых точек $\boldsymbol{z}', \boldsymbol{z}'' \in H_c(\boldsymbol{z}),\ \boldsymbol{z}' \neq \boldsymbol{z}''$ справедливо

$$\langle \boldsymbol{c}, \boldsymbol{\gamma}_i(\boldsymbol{z}') \rangle < \langle \boldsymbol{c}, \boldsymbol{\gamma}_i(\boldsymbol{z}'') \rangle \Leftrightarrow \beta_i(\boldsymbol{z}') < \beta_i(\boldsymbol{z}'')$$

для всех $i \in \mathcal{P}$.

Доказательство. Поскольку $\boldsymbol{z}', \boldsymbol{z}'' \in H_c(\boldsymbol{z})$, а $\boldsymbol{c}$ является нормалью к гиперплоскости $H_c(\boldsymbol{z})$, справедливы следующие два равенства

$$\langle \boldsymbol{c}, \boldsymbol{z}' - \boldsymbol{z} \rangle = 0;$$

$$\langle \boldsymbol{c}, \boldsymbol{z}'' - \boldsymbol{z} \rangle = 0.$$

Следовательно

$$\langle \boldsymbol{c}, \boldsymbol{z}'' \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle = \langle \boldsymbol{c}, \boldsymbol{z}' \rangle . \tag{17}$$

Последовательно используя формулы (14), (17) и (13), получаем следующую цепочку эквивалентных неравенств:

$$\begin{aligned}
&\langle \boldsymbol{c}, \boldsymbol{\gamma}_i(\boldsymbol{z}') \rangle < \langle \boldsymbol{c}, \boldsymbol{\gamma}_i(\boldsymbol{z}'') \rangle \Leftrightarrow \langle \boldsymbol{c}, \boldsymbol{z}' + \beta_i(\boldsymbol{z}')\boldsymbol{e_c} \rangle < \langle \boldsymbol{c}, \boldsymbol{z}'' + \beta_i(\boldsymbol{z}'')\boldsymbol{e_c} \rangle \\
&\Leftrightarrow \langle \boldsymbol{c}, \boldsymbol{z}' \rangle + \langle \boldsymbol{c}, \beta_i(\boldsymbol{z}')\boldsymbol{e_c} \rangle < \langle \boldsymbol{c}, \boldsymbol{z}'' \rangle + \langle \boldsymbol{c}, \beta_i(\boldsymbol{z}'')\boldsymbol{e_c} \rangle \\
&\Leftrightarrow \langle \boldsymbol{c}, \beta_i(\boldsymbol{z}')\boldsymbol{e_c} \rangle < \langle \boldsymbol{c}, \beta_i(\boldsymbol{z}'')\boldsymbol{e_c} \rangle \\
&\Leftrightarrow \langle \boldsymbol{c}, \beta_i(\boldsymbol{z}')\boldsymbol{c}/\left\|\boldsymbol{c}\right\| \rangle < \langle \boldsymbol{c}, \beta_i(\boldsymbol{z}'')\boldsymbol{c}/\left\|\boldsymbol{c}\right\| \rangle \\
&\Leftrightarrow \frac{\beta_i(\boldsymbol{z}')}{\|\boldsymbol{c}\|} \langle \boldsymbol{c}, \boldsymbol{c} \rangle < \frac{\beta_i(\boldsymbol{z}'')}{\|\boldsymbol{c}\|} \langle \boldsymbol{c}, \boldsymbol{c} \rangle \\
&\Leftrightarrow \beta_i(\boldsymbol{z}') < \beta_i(\boldsymbol{z}'').
\end{aligned}$$

Утверждение доказано. □

Следуя [39] дадим определение рецессивного полупространства.

Определение 4. Полупространство $\hat{H}_i$ называется рецессивным, если

$$\forall \boldsymbol{x} \in H_i, \forall \lambda > 0 : \boldsymbol{x} + \lambda \boldsymbol{c} \notin \hat{H}_i. \tag{18}$$

Геометрический смысл этого определения состоит в том, что луч, исходящий в направлении вектора $\boldsymbol{c}$ из любой точки гиперплоскости, ограничивающей рецессивное полупространство, не имеет общих точек с этим полупространством, за исключением начальной. Известно [39], что следующее условие является необходимым и достаточным для того, чтобы полупространство $\hat{H}_i$ было рецессивным:

$$\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle > 0. \tag{19}$$

Рецессивное полупространство обладает следующими свойствами.

Свойство 1. Пусть полупространство $\hat{H}_i$ является рецессивным. Тогда любая прямая, параллельная вектору $\boldsymbol{c}$, пересекает гиперплоскость H_i в единственной точке.

Данное свойство непосредственно вытекает из того факта, что гиперплоскость H_i, ограничивающая рецессивное полупространство $\hat{H}_i$ по определению не может быть параллельна вектору $\boldsymbol{c}$.

Свойство 2. Пусть полупространство $\hat{H}_i$ является рецессивным. Тогда

$$\boldsymbol{x} \in \hat{H}_i \Leftrightarrow \beta_i(\boldsymbol{x}) \geqslant 0. \tag{20}$$

Доказательство. Сначала предположим, что $\boldsymbol{x} \in \hat{H}_i$. Тогда в соответствии с (2) справедливо неравенство

$$\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle - b_i \leqslant 0.$$

В силу (12) имеем

$$\beta_i(\boldsymbol{x}) = -\frac{\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle - b_i}{\langle \boldsymbol{a}_i, \boldsymbol{c} \rangle} \left\|\boldsymbol{c}\right\| . \tag{21}$$

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Принимая во внимание (19), получаем

$$\boldsymbol{x} \in \hat{H}_i \Rightarrow \beta_i(\boldsymbol{x}) \geqslant 0.$$

Теперь предположим, что $\beta_i(\boldsymbol{x}) \geqslant 0$. В соответствии с (21) это означает, что

$$\frac{\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle - b_i}{\langle \boldsymbol{a}_i, \boldsymbol{c}\rangle} \|\boldsymbol{c}\| \leqslant 0.$$

Учитывая (19), отсюда получаем

$$\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle - b_i \leqslant 0.$$

Согласно (2), отсюда следует, что

$$\boldsymbol{x} \in \hat{H}_i.$$

Таким образом

$$\beta_i(\boldsymbol{x}) \geqslant 0 \Rightarrow \boldsymbol{x} \in \hat{H}_i.$$

Свойство доказано. □

Определим

$$\mathcal{I} = \{ i \in \mathcal{P} \,|\, \langle \boldsymbol{a}_i, \boldsymbol{c}\rangle > 0 \}, \tag{22}$$

то есть $\mathcal{I}$ представляет множество индексов, для которых полупространство $\hat{H}_i$ является рецессивным. Поскольку допустимый многогранник M представляет собой ограниченное множество, имеем

$$\mathcal{I} \neq \emptyset. \tag{23}$$

Положим

$$\hat{M} = \bigcap_{i \in \mathcal{I}} \hat{H}_i. \tag{24}$$

Очевидно, что $\hat{M}$ является выпуклым, замкнутым, неограниченным многогранником. Будем называть его рецессивным. Из (4) и (22) следует

$$M \subset \hat{M}. \tag{25}$$

Обозначим через $\Gamma(\hat{M})$ множество граничных точек рецессивного многогранника $\hat{M}$. Согласно утверждению 3 в [39] имеем

$$\bar{\boldsymbol{x}} \in \Gamma(\hat{M}),$$

то есть решение задачи ЛП (1) лежит на границе рецессивного многогранника $\hat{M}$.

Утверждение 4. Пусть $\hat{M}$ — рецессивный многогранник, определяемый формулой (24). Тогда для любой точки $\boldsymbol{u} \in \Gamma(\hat{M})$ существует $\epsilon > 0$ такой, что для любой граничной точки $\boldsymbol{w}$, принадлежащей ϵ-окрестности $V_\epsilon(\boldsymbol{u})$ точки $\boldsymbol{u}$, найдется $i' \in \mathcal{I}$, для которого справедливо $\boldsymbol{u}, \boldsymbol{w} \in H_{i'}$:

$$\forall \boldsymbol{u} \in \Gamma(\hat{M})\ \exists \epsilon > 0 : \forall \boldsymbol{w} \in V_\epsilon(\boldsymbol{u}) \cap \Gamma(\hat{M})\ \exists i' \in \mathcal{I} : \boldsymbol{u}, \boldsymbol{w} \in H_{i'}.$$

Доказательство. Доказательство идентично доказательству утверждения 1. □

Определение 5. Целевой проекцией точки $\boldsymbol{z} \in \mathbb{R}^n$ на границу $\Gamma(\hat{M})$ рецессивного многогранника $\hat{M}$ называется точка $\hat{\boldsymbol{\gamma}}(\boldsymbol{z})$, вычисляемая по формуле

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) = L(\boldsymbol{z}) \cap \Gamma(\hat{M}),$$

где $L(\boldsymbol{z})$ — прямая, проходящую через точку $\boldsymbol{z}$ параллельно вектору $\boldsymbol{c}$:

$$L(\boldsymbol{z}) = \{ \boldsymbol{y} \in \mathbb{R}^n | \, \boldsymbol{y} = \boldsymbol{z} + \lambda \boldsymbol{c}, \lambda \in \mathbb{R} \}.$$

Скалярную величину $\hat{\beta}(\boldsymbol{z}) \in \mathbb{R}$, удовлетворяющую уравнению

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) = \boldsymbol{z} + \hat{\beta}(\boldsymbol{z})\boldsymbol{c} \tag{26}$$

будем называть смещением точки $\boldsymbol{z}$ относительно границы рецессивного многогранника $\hat{M}$.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Заметим, что корректность этого определения базируется на свойстве 1. Следующее утверждение предоставляет формулу для вычисления целевой проекции на границу рецессивного многогранника.

Утверждение 5. Пусть задана произвольная точка $\boldsymbol{z} \in \mathbb{R}^n$. Положим

$$i' = \arg\min \{\beta_i(\boldsymbol{z}) \,|\, i \in \mathcal{I}\}. \tag{27}$$

Тогда

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) = \boldsymbol{\gamma}_{i'}(\boldsymbol{z}). \tag{28}$$

Другими словами, целевая проекция точки $\boldsymbol{z}$ на границу рецессивного многогранника $\hat{M}$ совпадает с проекцией этой точки на гиперплоскость $H_{i'}$, имеющей минимальное смещение относительно $\boldsymbol{z}$.

Доказательство. Зафиксируем произвольную точку $\boldsymbol{z} \in \mathbb{R}^n$. В соответствии с определением 5 построим прямую, параллельную вектору $\boldsymbol{c}$, которая проходит через точку $\boldsymbol{z}$:

$$L = \{\boldsymbol{y} \in \mathbb{R}^n \,|\, \boldsymbol{y} = \boldsymbol{z} + \lambda \boldsymbol{c}, \lambda \in \mathbb{R}\}.$$

Имеем

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) = L \cap \Gamma(\hat{M}).$$

Согласно свойству 1 для любого $i \in \mathcal{I}$ прямая L пересекает H_i только в одной точке, которую мы обозначим через $\boldsymbol{y}_i$:

$$L \cap H_i = \{\boldsymbol{y}_i\}.$$

Определим

$$Y = \bigcup_{i \in \mathcal{I}} \{\boldsymbol{y}_i\},$$

то есть Y — множество точек, в которых прямая L пересекает границы рецессивных полупространств. По определению 1 имеем

$$\forall i \in \mathcal{I} \;:\; \boldsymbol{\gamma}_i(\boldsymbol{z}) = \boldsymbol{y}_i, \tag{29}$$

и

$$\forall i \in \mathcal{I} \;:\; \boldsymbol{y}_i \in H_i. \tag{30}$$

В силу (24) также имеем

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) \in Y.$$

Это означает, что найдется $i' \in \mathcal{I}$ такой, что

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) = \boldsymbol{\gamma}_{i'}(\boldsymbol{z}). \tag{31}$$

Покажем, что

$$i' \in \operatorname{Arg\,min} \{\beta_i(\boldsymbol{z}) \,|\, i \in \mathcal{I}\}.$$

Предположим противное, то есть

$$\beta_{i'}(\boldsymbol{z}) > \min \{\beta_i(\boldsymbol{z}) \,|\, i \in \mathcal{I}\}.$$

Тогда существует $i'' \in \mathcal{I}$ такой, что

$$\beta_{i''}(\boldsymbol{z}) < \beta_{i'}(\boldsymbol{z}) \tag{32}$$

В силу (14) и (29) имеем

$$\boldsymbol{y}_{i'} = \boldsymbol{z} + \beta_{i'}(\boldsymbol{z}) \boldsymbol{e}_{\boldsymbol{c}};$$

$$\boldsymbol{y}_{i''} = \boldsymbol{z} + \beta_{i''}(\boldsymbol{z}) \boldsymbol{e}_{\boldsymbol{c}}.$$

Отсюда

$$\boldsymbol{y}_{i'} = \boldsymbol{y}_{i''} + (\beta_{i'}(\boldsymbol{z}) - \beta_{i''}(\boldsymbol{z}))\, \boldsymbol{e}_{\boldsymbol{c}},$$

что в силу (32) и (13) равносильно

$$\boldsymbol{y}_{i'} = \boldsymbol{y}_{i''} + \frac{|\beta_{i''}(\boldsymbol{z}) - \beta_{i'}(\boldsymbol{z})|}{\|\boldsymbol{c}\|} \boldsymbol{c}. \tag{33}$$

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

В соответствии с (18) и (30), из (33) следует

$$\boldsymbol{y}_{i'} \notin \hat{H}_{i''}.$$

Это означает, что

$$\boldsymbol{y}_{i'} \notin \hat{M}.$$

Принимая во внимание (29) и (31), отсюда следует

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) \notin \hat{M}.$$

Получили противоречие с определением 5. *Утверждение доказано.* □

Следующее утверждение предоставляет формулу для вычисления смещения точки относительно границы рецессивного многогранника.

Утверждение 6. Пусть задана произвольная точка $\boldsymbol{z} \in \mathbb{R}^n$. Тогда

$$\hat{\beta}(\boldsymbol{z}) = \frac{\langle \boldsymbol{c}, \hat{\boldsymbol{\gamma}}(\boldsymbol{z}) - \boldsymbol{z} \rangle}{\|\boldsymbol{c}\|^2}. \tag{34}$$

Доказательство. В соответствии с (26) точки $\hat{\boldsymbol{\gamma}}(\boldsymbol{z})$ и $\boldsymbol{z}$ находятся на нормали к гиперплоскости $H_c(\boldsymbol{z})$. Поэтому точка $\boldsymbol{z} \in H_c(\boldsymbol{z})$ является ортогональной проекцией точки $\hat{\boldsymbol{\gamma}}(\boldsymbol{z})$ на гиперплоскость $H_c(\boldsymbol{z})$, и с учетом (16) может быть вычислена по известной формуле

$$\boldsymbol{z} = \hat{\boldsymbol{\gamma}}(\boldsymbol{z}) - \frac{\langle \boldsymbol{c}, \hat{\boldsymbol{\gamma}}(\boldsymbol{z}) - \boldsymbol{z} \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c}. \tag{35}$$

Перепишем это в виде

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) = \boldsymbol{z} + \frac{\langle \boldsymbol{c}, \hat{\boldsymbol{\gamma}}(\boldsymbol{z}) - \boldsymbol{z} \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c}.$$

Сопоставляя это с (26), получаем

$$\hat{\beta}(\boldsymbol{z}) = \frac{\langle \boldsymbol{c}, \hat{\boldsymbol{\gamma}}(\boldsymbol{z}) - \boldsymbol{z} \rangle}{\|\boldsymbol{c}\|^2}.$$

Утверждение доказано. □

3. Метод поверхностного движения. Метод поверхностного движения строит на поверхности допустимого многогранника путь из произвольной граничной точки $\boldsymbol{u}^{(0)} \in M \cap \Gamma(\hat{M})$ до точки $\bar{\boldsymbol{x}}$, являющейся решением задачи ЛП (1). Перемещение по поверхности рецессивного многогранника происходит в направлении наибольшего увеличения значения целевой функции. Реализация метода поверхностного движения приведена в виде алгоритма 1. Прокомментируем шаги этого алгоритма. На шаге 1 вводится начальное приближение $\boldsymbol{u}^{(0)}$. Это может быть произвольная граничная точка рецессивного многогранника $\hat{M}$, удовлетворяющая условию

$$\boldsymbol{u}^{(0)} \in M \cap \Gamma(\hat{M}).$$

Для получения хорошего начального приближения может применяться алгоритм, реализующий стадию Quest апекс-метода [39]. На шаге 2 счетчик итераций k устанавливается в 0 и задается начальное значение параметра r. На шаге 3 строится n-мерный диск D, являющийся пересечением целевой гиперплоскости $H_c\left(\boldsymbol{u}^{(0)}\right)$, проходящей через точку $\boldsymbol{u}^{(0)}$, и n-мерного шара $V_r\left(\boldsymbol{u}^{(0)}\right)$ малого радиуса r с центром в точке $\boldsymbol{u}^{(0)}$. На шаге 4 вычисляется точка $\boldsymbol{v} \in D$ с максимальным смещением относительно границы рецессивного многогранника $\hat{M}$. Смещение $\hat{\beta}(\boldsymbol{z})$ вычисляется с помощью формулы (34). Целевая проекция $\hat{\boldsymbol{\gamma}}(\boldsymbol{z})$, используемая в формуле (34), вычисляется с помощью формул (27) и (28). Смещение $\beta_i(\boldsymbol{z})$, используемое в формуле (27), вычисляется с помощью формулы (12). Шаг 5 вычисляет точку $\boldsymbol{w}$, являющуюся целевой проекцией точки $\boldsymbol{v}$ на границу рецессивного многогранника. Шаги 6–15 реализуют основной аппроксимирующий цикл метода поверхностного движения, геометрическая интерпретация которого приведена на рис. 2. Этот цикл выполняется пока справедливо условие

$$\left\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{(k)} \right\rangle > \epsilon_f, \tag{36}$$

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Алгоритм 1. Метод поверхностного движения

Algorithm 1. Surface movement method

Require $\hat{H}_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \leqslant b_i\}$, $\hat{M} = \bigcap\limits_{i \in \mathcal{I}} \hat{H}_i$, $H_c(\boldsymbol{z}) = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{c}, \boldsymbol{x} - \boldsymbol{z} \rangle = 0\}$

1: **input** $\boldsymbol{u}^{(0)}$; **assert** $\boldsymbol{u}^{(0)} \in M \cap \Gamma(\hat{M})$
2: $k := 0; r := 0.1$
3: $D := H_c(\boldsymbol{u}^{(0)}) \cap V_r(\boldsymbol{u}^{(0)})$
4: $\boldsymbol{v} := \arg\max\{\hat{\beta}(\boldsymbol{x}) \mid \boldsymbol{x} \in D\}$
5: $\boldsymbol{w} := \hat{\boldsymbol{\gamma}}(\boldsymbol{v})$
6: **while** $\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{(k)} \rangle > \epsilon_f$ **do**
7: **assert** $\exists i \in \mathcal{I} : \boldsymbol{w}, \boldsymbol{u}^{(k)} \in H_i$ // Если не выполняется, уменьшить r
8: $\boldsymbol{d} := \boldsymbol{w} - \boldsymbol{u}^{(k)}$
9: $L = \{\boldsymbol{u}^{(k)} + \lambda \boldsymbol{d} \mid \lambda \in \mathbb{R}_{>0}\}$
10: $\boldsymbol{u}^{(k+1)} := \arg\max\{\|\boldsymbol{x} - \boldsymbol{u}^{(k)}\| \mid \boldsymbol{x} \in L \cap \Gamma(M)\}$
11: $D := H_c(\boldsymbol{u}^{(k+1)}) \cap V_r(\boldsymbol{u}^{(k+1)})$
12: $\boldsymbol{v} := \arg\max\{\hat{\beta}(\boldsymbol{x}) \mid \boldsymbol{x} \in D\}$
13: $\boldsymbol{w} := \hat{\boldsymbol{\gamma}}(\boldsymbol{v})$
14: $k := k + 1$
15: **end while**
16: **output** $\boldsymbol{u}^{(k)}$
17: **stop**

где ϵ_f — малый положительный параметр. Шаг 7 проверяет, что существует рецессивное полупространство $\hat{H}_i$ такое, что граничные точки $\boldsymbol{w}$ и $\boldsymbol{u}^{(k)}$ лежат на гиперплоскости H_i ограничивающей данное полупространство. Это необходимо для того, чтобы перемещение происходило по поверхности рецессивного многогранника, а не через его внутреннюю область. Если указанное требование не выполняется, необходимо уменьшить радиус r n-мерного шара $V_r\left(\boldsymbol{u}^{(k)}\right)$. Подходящий r найдется в силу утверждения 4. На шаге 8 формируется вектор $\boldsymbol{d}$, определяющий направление движения:

$$\boldsymbol{d} = \boldsymbol{w} - \boldsymbol{u}^{(k)}.$$

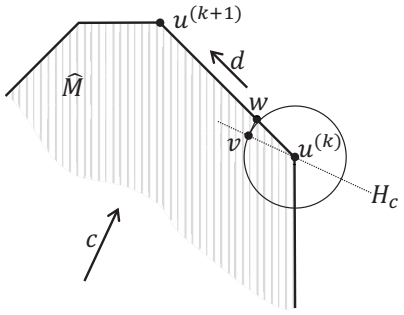

Рис. 2. Итерация основного цикла в методе поверхностного движения

Fig. 2. Iteration of main loop of surface motion method

Шаг 9 строит луч L с началом в точке $\boldsymbol{u}^{(k)}$, сонаправленный с вектором $\boldsymbol{d}$. На шаге 10 определяется следующее приближение $\boldsymbol{u}^{(k+1)}$ как точку на луче L, лежащую на границе допустимого многогранника M и максимально удаленную от точки $\boldsymbol{u}^{(k)}$. По построению из (36) следует

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(k)} \right\rangle < \langle \boldsymbol{c}, w \rangle \leqslant \left\langle \boldsymbol{c}, \boldsymbol{u}^{(k+1)} \right\rangle. \tag{37}$$

Шаг 11 строит новый гипердиск D радиуса r с центром в $\boldsymbol{u}^{(k+1)}$:

$$D := H_c\left(\boldsymbol{u}^{(k+1)}\right) \cap V_r\left(\boldsymbol{u}^{(k+1)}\right).$$

Шаг 12 находит на гипердиске D точку $\boldsymbol{v}$ с максимальным смещением. На шаге 13 вычисляется точка $\boldsymbol{w}$, являющаяся целевой проекцией точки $\boldsymbol{v}$ на границу рецессивного многогранника $\hat{M}$. Шаг 14 увеличивает счетчик итераций k на единицу. На шаге 15 происходит переход на начало основного цикла **while**. Шаг 16 выводит в качестве результата последнее приближение $\boldsymbol{u}^{(k)}$. Шаг 17 завершает работу алгоритма. Отметим, что по построению алгоритма 1 для любого k имеет место

$$\boldsymbol{u}^{(k)} \in M \cap \Gamma(\hat{M}), \tag{38}$$

то есть все точки последовательности $\left\{\boldsymbol{u}^{(k)}\right\}$, генерируемой алгоритмом 1, одновременно лежат и на границе допустимого многогранника M, и на границе рецессивного многогранника $\hat{M}$.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Следующая лемма гарантирует завершение работы алгоритма 1 за конечное число итераций.

Лемма 1. Пусть допустимый многогранник M задачи ЛП (1) является ограниченным непустым множеством. Тогда последовательность точек $\left\{\boldsymbol{u}^{(k)}\right\}$, генерируемая алгоритмом 1, является конечной для любого $\epsilon_f > 0$.

Доказательство. Предположим противное, то есть алгоритм 1 генерирует бесконечную последовательность точек $\left\{\boldsymbol{u}^{(k)}\right\}$. Но тогда, в силу (37), мы получаем бесконечную строго возрастающую числовую последовательность

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(0)} \right\rangle < \left\langle \boldsymbol{c}, \boldsymbol{u}^{(1)} \right\rangle < \left\langle \boldsymbol{c}, \boldsymbol{u}^{(2)} \right\rangle < \ldots \tag{39}$$

Так как по условию леммы допустимый многогранник M является непустым ограниченным множеством, задачи ЛП (1) имеет решение $\bar{\boldsymbol{x}}$. В силу (38) имеем

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(k)} \right\rangle \leqslant \langle \boldsymbol{c}, \bar{\boldsymbol{x}} \rangle$$

для всех $k = 0, 1, 2, \ldots$ Это означает, что последовательность (39) является ограниченной сверху. По теореме Вейерштрасса монотонно возрастающая ограниченная сверху числовая последовательность имеет конечный предел, равный ее супремуму. То есть существует $k' \in \mathbb{N}$ такой, что

$$\forall k > k' : \left\langle \boldsymbol{c}, \boldsymbol{u}^{(k+1)} \right\rangle - \left\langle \boldsymbol{c}, \boldsymbol{u}^{(k)} \right\rangle < \epsilon_f.$$

В силу (37) отсюда следует

$$\forall k > k' : \langle \boldsymbol{c}, \boldsymbol{w} \rangle - \left\langle \boldsymbol{c}, \boldsymbol{u}^{(k)} \right\rangle < \epsilon_f,$$

что равносильно

$$\forall k > k' : \left\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{(k)} \right\rangle < \epsilon_f.$$

Получили противоречие с условием (36) выполнения цикла, используемом на шаге 6 алгоритма 1. *Лемма доказана.* □

Следующая теорема показывает, что при достаточно малом ϵ_f результатом работы алгоритма 1 будет решение задачи ЛП (1).

Теорема 1. Пусть допустимый многогранник M задачи ЛП (1) является ограниченным непустым множеством. Пусть $\bar{\boldsymbol{x}}$ — решение задачи ЛП (1). Пусть задана монотонно убывающая последовательность положительных чисел $\{\epsilon_K\}_{K=1}^{\infty}$, стремящаяся к нулю:

$$\lim_{K \to \infty} \epsilon_K = 0. \tag{40}$$

Обозначим через $u^{(K)}$ конечную точку, генерируемую алгоритмом 1 при $\epsilon_f = \epsilon_K$ (она существует в силу леммы 1). Тогда найдется $\bar{K} \in \mathbb{N}$ такое, что для всех $K \geqslant \bar{K}$

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle = \langle \boldsymbol{c}, \bar{\boldsymbol{x}} \rangle .$$

Доказательство. Сначала покажем, что последовательность $\left\{\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle\right\}_{K=1}^{\infty}$ является монотонно возрастающей. Зафиксируем произвольное $K \in \mathbb{N}$ и предположим противное, то есть

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle > \left\langle \boldsymbol{c}, \boldsymbol{u}^{(K+1)} \right\rangle,$$

что равносильно

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle - \left\langle \boldsymbol{c}, \boldsymbol{u}^{(K+1)} \right\rangle > 0. \tag{41}$$

Так как $\boldsymbol{u}^{(K)}$ является конечной точкой при $\epsilon_f = \epsilon_K$, то в соответствии с шагом 6 алгоритма 1 имеем

$$\left\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{(K)} \right\rangle \leqslant \epsilon_K. \tag{42}$$

Аналогично имеем

$$\left\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{(K+1)} \right\rangle \leqslant \epsilon_{K+1}. \tag{43}$$

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Вычтя (42) из (43) и приведя подобные, получаем

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle - \left\langle \boldsymbol{c}, \boldsymbol{u}^{(K+1)} \right\rangle \leqslant \epsilon_{K+1} - \epsilon_K.$$

По условию теоремы

$$\epsilon_{K+1} \leqslant \epsilon_K.$$

Следовательно справедливо неравенство

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle - \left\langle \boldsymbol{c}, \boldsymbol{u}^{(K+1)} \right\rangle \leqslant 0.$$

Получили противоречие с (41). Значит последовательность $\left\{\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle\right\}_{K=1}^{\infty}$ является монотонно возрастающей. Очевидно, что эта последовательность ограничена сверху величиной $\langle \boldsymbol{c}, \bar{\boldsymbol{x}} \rangle$. Следовательно она имеет конечный предел:

$$\lim_{K \to \infty} \left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle = \bar{f}.$$

Алгоритм 1 в рамках каждой итерации проходит[3] одну грань/ребро рецессивного многогранника $\hat{M}$ в направлении максимального увеличения значения целевой функции. При этом каждая грань/ребро проходится не более одного раза, так как многогранник $\hat{M}$ является выпуклым множеством. Это означает, что существует $\bar{K} \in \mathbb{N}$ такое, что для всех $K \geqslant \bar{K}$ имеем

$$\boldsymbol{u}^{(K)} = \boldsymbol{u}^{\left(\bar{K}\right)}$$

и

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{(K)} \right\rangle = \bar{f}.$$

По построению алгоритма 1, учитывая (40), это возможно только в том случае, когда

$$\left\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{\left(\bar{K}\right)} \right\rangle = 0. \tag{44}$$

Покажем, что в этом случае

$$\left\langle \boldsymbol{c}, \boldsymbol{u}^{\left(\bar{K}\right)} \right\rangle = \langle \boldsymbol{c}, \bar{\boldsymbol{x}} \rangle,$$

то есть точка $\boldsymbol{u}^{\left(\bar{K}\right)}$ является решением задачи ЛП (1). Обозначим $\boldsymbol{u}' = \boldsymbol{u}^{\left(\bar{K}\right)}$. Предположим противное: существует точка

$$\boldsymbol{u}'' \in M \tag{45}$$

такая, что

$$\langle \boldsymbol{c}, \boldsymbol{u}'' \rangle > \langle c, \boldsymbol{u}' \rangle.$$

Это равносильно

$$\langle \boldsymbol{c}, \boldsymbol{u}'' - \boldsymbol{u}' \rangle > 0. \tag{46}$$

Рис. 3 иллюстрирует последующую часть доказательства. Исходя из определения 3, вычислим ортогональную проекцию $\boldsymbol{p}$ точки $\boldsymbol{u}''$ на целевую гиперплоскость $H_c(\boldsymbol{u}')$, проходящую через точку $\boldsymbol{u}'$:

$$\boldsymbol{p} = \boldsymbol{u}'' - \frac{\langle \boldsymbol{c}, \boldsymbol{u}'' - \boldsymbol{u}' \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c}. \tag{47}$$

Заметим, что

$$\|\boldsymbol{p} - \boldsymbol{u}'\| \neq 0, \tag{48}$$

так как в противном случае, в соответствии с определением 4, точка $\boldsymbol{u}''$ не может принадлежать рецессивному многограннику $\hat{M}$, что противоречит формуле (45). Выберем $r \in \mathbb{R}$, удовлетворяющий условию

$$r > 0, \tag{49}$$

[3] Под прохождением грани/ребра многогранника понимается движение внутри линейного многообразия размерности k при наличии k степеней свободы ($0 < k < n$).

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

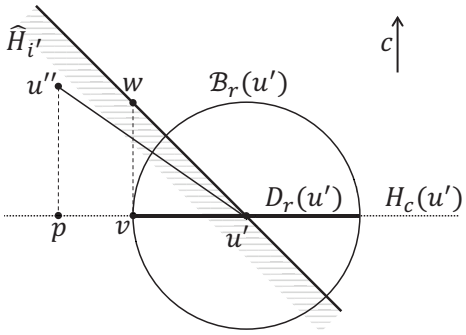

Рис. 3. Иллюстрация к доказательству теоремы 1

Fig. 3. Illustration to proof of Theorem 1

для которого существует $i' \in \mathcal{I}$ такой, что

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{v}), \boldsymbol{u}' \in H_{i'} \cap \Gamma(M), \tag{50}$$

где

$$\boldsymbol{v} = \boldsymbol{u}' + \frac{r}{\|\boldsymbol{p} - \boldsymbol{u}'\|}(\boldsymbol{p} - \boldsymbol{u}'). \tag{51}$$

Это возможно в силу утверждения 1. Отметим, что

$$i' = \arg\min\left\{\beta_i(\boldsymbol{v}) \,|\, i \in \mathcal{I}\right\},$$

так как в противном случае $\hat{\boldsymbol{\gamma}}(\boldsymbol{v}) \notin H_{i'}$, что противоречит (50). Согласно утверждению 5 отсюда следует

$$\hat{\boldsymbol{\gamma}}(\boldsymbol{v}) = \boldsymbol{\gamma}_{i'}(\boldsymbol{v}). \tag{52}$$

В соответствии с шагом 13 алгоритма 1 положим

$$\boldsymbol{w} = \hat{\boldsymbol{\gamma}}(\boldsymbol{v}).$$

С учетом (52) имеем

$$\boldsymbol{w} = \boldsymbol{\gamma}_{i'}(\boldsymbol{v}).$$

Используя утверждение 2 отсюда получаем

$$\boldsymbol{w} = \boldsymbol{v} - \frac{\langle \boldsymbol{a}_{i'}, \boldsymbol{v}\rangle - b_{i'}}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c}\rangle}\boldsymbol{c}. \tag{53}$$

Поскольку $\boldsymbol{u}' \in H_{i'}$, из (3) следует

$$\langle \boldsymbol{a}_{i'}, \boldsymbol{u}'\rangle = b_{i'}. \tag{54}$$

Поэтому (53) можно переписать в виде

$$\boldsymbol{w} = \boldsymbol{v} - \frac{\langle \boldsymbol{a}_{i'}, \boldsymbol{v}\rangle - \langle \boldsymbol{a}_{i'}, \boldsymbol{u}'\rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c}\rangle}\boldsymbol{c}.$$

Подставив вместо $\boldsymbol{v}$ правую часть формулы (51), отсюда получаем

$$\boldsymbol{w} = \boldsymbol{u}' + \frac{r}{\|\boldsymbol{p} - \boldsymbol{u}'\|}(\boldsymbol{p} - \boldsymbol{u}') - \frac{\left\langle \boldsymbol{a}_{i'}, \boldsymbol{u}' + \frac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|}(\boldsymbol{p} - \boldsymbol{u}')\right\rangle - \langle \boldsymbol{a}_{i'}, \boldsymbol{u}'\rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c}\rangle}\boldsymbol{c},$$

что равносильно

$$\boldsymbol{w} = \boldsymbol{u}' + \frac{r}{\|\boldsymbol{p} - \boldsymbol{u}'\|}(\boldsymbol{p} - \boldsymbol{u}') - \frac{\left\langle \boldsymbol{a}_{i'}, \frac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|}(\boldsymbol{p} - \boldsymbol{u}')\right\rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c}\rangle}\boldsymbol{c}. \tag{55}$$

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

В соответствии с (2) имеем

$$\hat{H}_{i'} = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_{i'}, \boldsymbol{x} \rangle \leqslant b_{i'}\}. \tag{56}$$

Используя (54), формулу (56) можно переписать в виде

$$\hat{H}_{i'} = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_{i'}, \boldsymbol{x} \rangle \leqslant \langle \boldsymbol{a}_{i'}, \boldsymbol{u}' \rangle\}. \tag{57}$$

Из (45) следует $\boldsymbol{u}'' \in \hat{H}_{i'}$. Сопоставляя это с (57) получаем

$$\langle \boldsymbol{a}_{i'}, \boldsymbol{u}'' \rangle \leqslant \langle \boldsymbol{a}_{i'}, \boldsymbol{u}' \rangle,$$

что равносильно

$$\langle \boldsymbol{a}_{i'}, \boldsymbol{u}' - \boldsymbol{u}'' \rangle \geqslant 0. \tag{58}$$

Поскольку полупространство $\hat{H}_{i'}$ является рецессивным, в соответствии с утверждением 1 в [39] справедливо

$$\langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle > 0. \tag{59}$$

В силу (55) и (47) имеем

$$\begin{aligned}
\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}' \rangle &= \left\langle \boldsymbol{c}, \tfrac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|}(\boldsymbol{p}-\boldsymbol{u}') - \frac{\left\langle \boldsymbol{a}_{i'}, \frac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|}(\boldsymbol{p}-\boldsymbol{u}') \right\rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle} \boldsymbol{c} \right\rangle = \tfrac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|} \left(\langle \boldsymbol{c}, \boldsymbol{p}-\boldsymbol{u}' \rangle - \tfrac{\langle \boldsymbol{a}_{i'}, \boldsymbol{p}-\boldsymbol{u}' \rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle} \|\boldsymbol{c}\| \right) \\
&= \tfrac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|} \left(\left\langle \boldsymbol{c}, \boldsymbol{u}'' - \tfrac{\langle \boldsymbol{c}, \boldsymbol{u}''-\boldsymbol{u}' \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c} - \boldsymbol{u}' \right\rangle - \frac{\left\langle \boldsymbol{a}_{i'}, \boldsymbol{u}'' - \frac{\langle \boldsymbol{c}, \boldsymbol{u}''-\boldsymbol{u}' \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c} - \boldsymbol{u}' \right\rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle} \|\boldsymbol{c}\| \right) \\
&= \tfrac{r}{\|\boldsymbol{p}-\boldsymbol{u}'\|} \left(- \frac{\left\langle \boldsymbol{a}_{i'}, \boldsymbol{u}'' - \frac{\langle \boldsymbol{c}, \boldsymbol{u}''-\boldsymbol{u}' \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c} - \boldsymbol{u}' \right\rangle}{\langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle} \|\boldsymbol{c}\| \right) = \tfrac{r\|\boldsymbol{c}\|}{\|\boldsymbol{p}-\boldsymbol{u}'\| \langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle} \left(- \left\langle \boldsymbol{a}_{i'}, \boldsymbol{u}'' - \tfrac{\langle \boldsymbol{c}, \boldsymbol{u}''-\boldsymbol{u}' \rangle}{\|\boldsymbol{c}\|^2} \boldsymbol{c} - \boldsymbol{u}' \right\rangle \right) \\
&= \tfrac{r\|\boldsymbol{c}\|}{\|\boldsymbol{p}-\boldsymbol{u}'\| \langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle} \left(\langle \boldsymbol{a}_{i'}, \boldsymbol{u}' - \boldsymbol{u}'' \rangle + \tfrac{\langle \boldsymbol{c}, \boldsymbol{u}''-\boldsymbol{u}' \rangle \langle \boldsymbol{a}_{i'}, \boldsymbol{c} \rangle}{\|\boldsymbol{c}\|^2} \right)
\end{aligned}$$

В соответствии с (49), (48), (59), (58) и (46) отсюда следует

$$\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}' \rangle > 0.$$

Вспомнив, что $\boldsymbol{u}' = \boldsymbol{u}^{(\bar{K})}$, перепишем последнее неравенство в виде

$$\left\langle \boldsymbol{c}, \boldsymbol{w} - \boldsymbol{u}^{(\bar{K})} \right\rangle > 0.$$

Получили противоречие с (44). *Теорема доказана.* □

4. Обсуждение. В данном разделе обсуждаются сильные и слабые стороны предложенного метода, а также раскрываются пути его практической реализации на основе синтеза суперкомпьютерных и нейросетевых технологий. Мы рассмотрим следующие вопросы.

1. Каковы пути реализации алгоритма 1 в виде программы для ЭВМ?
2. Что можно сказать о временной сложности алгоритма?
3. Может ли метод поверхностного движения применяться к нестационарным задачам ЛП?
4. Какое отношение метод поверхностного движения имеет к проблеме реального времени?
5. Является ли путь, строящийся в методе поверхностного движения, кратчайшим?
6. В чем слабые стороны предложенного метода?
7. В чем заключается научная новизна и значимость метода поверхностного движения?

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Алгоритм 2. Метод ЛП с использованием DNN

Algorithm 2. LP method using DNN

Require $\hat{H}_i = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \leqslant b_i\}$, $\hat{M} = \bigcap\limits_{i \in \mathcal{I}} \hat{H}_i$, $H_c(\boldsymbol{z}) = \{\boldsymbol{x} \in \mathbb{R}^n | \langle \boldsymbol{c}, \boldsymbol{x} - \boldsymbol{z} \rangle = 0\}$

1: **input** $\boldsymbol{u}^{(0)}$; **assert** $\boldsymbol{u}^{(0)} \in M \cap \Gamma(\hat{M})$
2: $k := 0$
3: $D := \mathfrak{G}(\boldsymbol{u}^{(0)})$
4: $\boldsymbol{d} := \mathrm{DNN}(D)$
5: **while** $\boldsymbol{d} \neq \boldsymbol{0}$ **do**
6: $L = \{\boldsymbol{u}^{(k)} + \lambda \boldsymbol{d} \mid \lambda \in \mathbb{R}_{>0}\}$
7: $\boldsymbol{u}^{(k+1)} := \arg\max\{\|\boldsymbol{x} - \boldsymbol{u}^{(k)}\| \mid \boldsymbol{x} \in L \cap \Gamma(M)\}$
8: $D := \mathfrak{G}(\boldsymbol{u}^{(k+1)})$
9: $\boldsymbol{d} := \mathrm{DNN}(D)$
10: $k := k + 1$
11: **end while**
12: **output** $\boldsymbol{u}^{(k)}$
13: **stop**

Начнем обсуждение с ответа на первый вопрос. На шагах 3 и 13 алгоритма 1 необходимо найти точку гипердиска D, имеющую максимальное смещение. Нам неизвестен алгоритм, позволяющий получить численное решение этой задачи. Однако мы видим следующий путь решения, предполагающий использование искусственной нейронной сети. Применяя подход, описанный в статье [37], мы заменяем гипердиск D на регулярное множество точек, называемых рецептивным полем. Каждой точке рецептивного поля мы сопоставляем ее смещение относительно границы допустимого многогранника. В результате мы получаем матрицу размерности $(n-1)$, представляющей собой локальный образ задачи ЛП. Локальность образа означает, что мы получаем визуальное представление поверхности не всего допустимого многогранника, а только некоторой его части в окрестности точки текущего приближения. Этот образ подается на вход предварительно обученной нейронной сети прямого распространения, которая определяет вектор $\boldsymbol{d}$, указывающий направление движения на поверхности допустимого многогранника в сторону максимального увеличения значения целевой функции. Обозначим через $\mathfrak{G}(\boldsymbol{u})$ функцию, строящую рецептивное поле с центром в точке $\boldsymbol{u}$ и вычисляющую локальный образ задачи ЛП в этой точке. Детально алгоритм построения многомерного образа задачи ЛП описан и исследован в работе [37]. Обозначим через DNN глубокую нейронную сеть прямого распространения, на вход которой подается локальный образ задачи ЛП, а на выходе получается вектор $\boldsymbol{d}$, задающий направление поверхностного движения. Тогда алгоритм 1 может быть преобразован в алгоритм 2, допускающий реализацию на практике. Множество размеченных прецедентов, необходимое для обучения DNN, может быть получено с помощью апекс-метода [39], строящего путь, близкий к оптимальному целевому пути[4].

Дадим оценку временной сложности алгоритма 2. Метод поверхностного движения посещает[5] каждую гиперплоскость рецессивного многогранника не более одного раза. Посещение гиперплоскости выполняется в пределах одной итерации цикла while (шаги 5-11 алгоритма 2). Следовательно, общее количество итераций можно оценить как $O(m)$, где m — количество ограничений задачи ЛП 1. Нахождение следующего приближения $\boldsymbol{u}^{(k+1)}$ можно реализовать путем дихотомии. Таким образом количество операций для шага 7 может быть оценено как $O(1)$, так как оно не зависит ни от количества ограничений m, ни от размерности n. Наиболее трудоемким является построение локального образа задачи ЛП на шаге 8 алгоритма 2. Рецептивное поле в виде гиперкубической решетки состоит из $\eta^{(n-1)}$ точек, где n — размерность пространства, а η — количество точек по одному измерению. Однако последние исследования [40] показали, что крестообразное рецептивное поле с количеством точек $(\eta - 1)n + 1$ дает результаты, не уступающие гиперкубическому по точности решения задачи ЛП. Предварительно обученная нейронная

[4]Оптимальный целевой путь — это путь на поверхности допустимого многогранника из начальной точки в направлении максимального увеличения значения целевой функции.

[5]Под посещением гиперплоскости понимается прямолинейное движение от точки входа на гиперплоскость до точки первого изменения направления движения.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

сеть прямого распространения DNN вычисляет вектор движения $\boldsymbol{d}$ на шаге 9 за время, зависящее только от n, так как на вход ей подается $(\eta-1)n+1$ чисел (в случае крестообразного рецептивного поля). Таким образом временная сложность алгоритма в целом может быть оценена как $O(mn)$. В дополнение отметим, что граница масштабируемости алгоритма построения визуального образа задачи ЛП может быть оценена как $O(\sqrt{2n^2m+m^2n+8nm-6m})$ [37]. Если предположить, что $m=O(n)$, то для границы масштабируемости[6] мы получаем оценку $O(n\sqrt{n})$, близкую к линейной зависимости. Это означает, что алгоритм построения локального образа задачи ЛП может быть эффективно распараллелен на большом количестве процессорных узлов кластерной вычислительной системы. Так для $n=7$ и $m=15$ вычислительные эксперимент показали пик ускорения на 326 процессорных узлах [37]. Заметим, что количество итераций алгоритма 2 не зависит от ϵ_f, так как такой параметр в этом алгоритме отсутствует.

Метод поверхностного движения является итерационным и самокорректирующимся. Поэтому он может потенциально применяться для решения нестационарных задач. При этом, если меняется только целевая функция, то алгоритм 2 вообще не требует никаких принципиальных изменений. Важно, чтобы скорость корректировки опережала скорость изменений. Если же меняется система ограничений (без изменения размерности), то алгоритм 2 потребует определенных модификаций, так как текущее приближение может "погрузиться"в многогранник или "оторваться"от его поверхности. Авторы планируют детально исследовать этот вопрос в будущем.

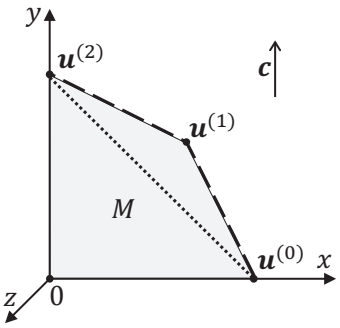

Рис. 4. Оптимальный целевой путь обозначен пунктиром; путь минимальной длины показан точками

Fig. 4. Optimal objective path is indicated by dashed line; path of minimum length is shown by dotted line

Алгоритм 2 также может применяться для решения задач ЛП в режиме реального времени. Действительно, количество итераций ограничено параметром m. При фиксированном n построение локального образа задачи ЛП требует фиксированного количества операций. При этом процедура построения образа эффективно распараллеливается на большом количестве процессорных узлов. Искусственная нейронная сеть прямого распространения DNN анализирует локальный образ задачи ЛП за фиксированное время, зависящее только от n. Работа нейронной сети также может быть эффективно распараллелена с помощью графических процессоров. В перспективе можно отказаться от путешествия по граням/ребрам допустимого многогранника и анализировать с помощью нейронной сети образ всего многогранника, полученный из точки апекса (см. [39]). Нейронная сеть будет выдавать приближенное решение, которое может быть уточнено путем анализа фиксированного числа локальных образов с увеличивающимся масштабом. Однако этот вопрос нуждается в дальнейших исследованиях.

Алгоритм 1 строит на поверхности допустимого многогранника оптимальный целевой путь к решению задачи ЛП. Это непосредственно следует из построения алгоритма и утверждения 3. Интересен вопрос, всегда ли этот путь является, путем наименьшей длины в смысле евклидовой метрики. Следующий простой пример в пространстве $\mathbb{R}^3$, проиллюстрированный на рис. 4, показывает, что это не так. Для системы ограничений

$$\begin{cases} x+2y\leqslant 2,\\ 2x+y\leqslant 2,\\ x\geqslant 0,\\ y\geqslant 0,\\ z=0; \end{cases}$$

и целевой функции $f(x,y,z)=y$ оптимальный целевой путь, обозначенный пунктиром, не совпадает с кратчайшим путем, обозначенным точками.

Также может возникнуть вопрос, можно ли на шаге 10 алгоритма 1 заменить $\Gamma(M)$ на $\Gamma(\hat{M})$, так как это сокращает количество неравенств, вовлекаемых в проверку условия $x\in\Gamma(M)$. Ответ — отрица-

[6] Под границей масштабируемости понимается число процессорных узлов кластерной вычислительной системы, на котором достигается максимум ускорения.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

тельный. Например, в пространстве $\mathbb{R}^2$ для системы ограничений

$$\begin{cases} x + 2y \leqslant 2, \\ 2x + y \leqslant 2, \\ x + y \geqslant 1, \\ x \geqslant 0, \\ y \geqslant 0; \end{cases}$$

и целевой функции $f(x, y) = y$ для $\boldsymbol{u}^{(1)} = \left(\frac{2}{3}; \frac{2}{3}\right)$ получаем

$$\arg\max\{\|\boldsymbol{x} - \boldsymbol{u}^{(1)}\| \mid \boldsymbol{x} \in L \cap \Gamma(\hat{M})\} = +\infty$$

(см. рис. 5).

В качестве недостатка метода поверхностного движения можно отметить то, что он аффинно не инвариантен, так как функционал цены отождествляется с вектором. Таким образом, поведение метода зависит от евклидовой структуры, определенной системой координат.

Научная новизна и теоретическая значимость предложенного метода заключается в том, что он впервые открывает возможность применения искусственных нейронных сетей прямого распространения для решения многомерных задач ЛП на основе анализа их образов.

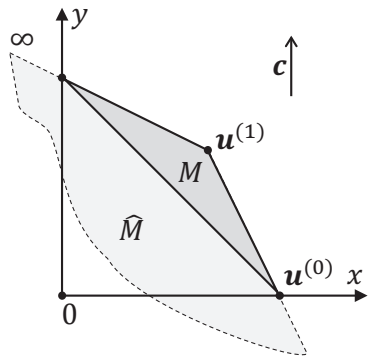

Рис. 5. Оптимальный целевой путь уходит из $\boldsymbol{u}^{(1)}$ в бесконечность на рецессивном многограннике $\hat{M}$

Fig. 5. Optimal objective path goes from $\boldsymbol{u}^{(1)}$ to infinity on recessive polytope $\hat{M}$

В настоящее время идет работа над реализацией представленного метода в виде программного комплекса для кластерных вычислительных систем. Этот программный комплекс включает в себя параллельный алгоритм генерации многомерных локальных образов задач ЛП, параллельный алгоритм генерации размеченных прецедентов для обучения нейронной сети и нейросетевые модели для различных размерностей. По завершении реализации мы планируем провести сравнение метода поверхностного движения с другими методами решения задач ЛП, в частности, с симплекс-методом. Описанию указанного программного комплекса и анализу результатов вычислительных экспериментов будет посвящена отдельная научная статья.

5. Заключение. В статье описан новый метод решения задачи линейного программирования (ЛП), получивший название “метод поверхностного движения”. Указанный метод строит на поверхности многогранника, ограничивающего допустимую область задачи ЛП, путь от начальной точки до точки решения задачи ЛП. Вектор движения всегда выбирается в направлении максимального увеличения/уменьшения значения целевой функции. Получившийся путь называется оптимальным целевым путем.

Метод поверхностного движения предполагает использование глубокой нейронной сети прямого распространения для определения направления движения по граням допустимого многогранника. Для этого строится многомерный локальный образ задачи ЛП в точке текущего приближения, который подается на вход нейронной сети. Множество размеченных прецедентов, необходимое для обучения нейронной сети может быть получено с помощью апекс-метода.

Для построения теоретического фундамента метода поверхностного движения введено понятие целевой проекции — косоугольной проекции в направлении, параллельном вектору градиента целевой функции. Определена скалярная величина, называемая смещением. Модуль смещения равен расстоянию от точки до ее целевой проекции. Знак смещения определяется положением точки внутри или вне допустимого многогранника. Получена формула вычисления смещения точки относительно границы допустимого многогранника. Показано, что большему целевому смещению соответствует большее значение целевой функции. Приведено формализованное описания метода поверхностного движения в виде алгоритма. Доказана основная теорема сходимости метода поверхностного движения к решению задачи ЛП за конечное число итераций. Приведен вариант алгоритма поверхностного движения, использующий функцию построения локального многомерного образа задачи ЛП и глубокую нейронную сеть.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

В качестве направлений дальнейших исследований можно указать следующие.

1. Разработка и обучение сети DNN, способной вычислять вектор движения в направлении максимального увеличения значения целевой функции для многомерных задач ЛП.

2. Разработка программного комплекса для кластерной вычислительной системы, реализующего алгоритм 2 путем синтеза суперкомпьютерных и нейросетевых технологий.

3. Исследование зависимости точности работы сети DNN от плотности рецептивного поля.

4. Исследование применимости метода поверхностного движения для решения нестационарных задач ЛП.

5. Исследование применимости метода поверхностного движения для решения задач ЛП в режиме реального времени.

6. Разработка и исследование нового визуального метода решения задач ЛП с помощью нейронных сетей на основе анализа образа допустимого многогранника в целом.

6. Обозначения.

$\mathbb{R}^n$	вещественное евклидово пространство
$\|\cdot\|$	евклидова норма
$\langle\cdot,\cdot\rangle$	скалярное произведение двух векторов
$[\cdot\,,\cdot]$	конкатенация двух векторов
$f(\boldsymbol{x})$	линейная целевая функция
$\boldsymbol{c}$	градиент целевой функции $f(\boldsymbol{x})$
$\boldsymbol{e_c}$	единичный вектор, сонаправленный с вектором $\boldsymbol{c}$
$\bar{\boldsymbol{x}}$	решение задачи ЛП
$\boldsymbol{a}_i$	i-тая строка матрицы A
H_i	гиперплоскость, определяемая формулой $\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle = b_i$
$\hat{H}_i$	полупространство, определяемое формулой $\langle \boldsymbol{a}_i, \boldsymbol{x}\rangle \leqslant b_i$
$\mathcal{P}$	множество индексов строк матрицы A
M	допустимый многогранник, определяемый формулой $M = \bigcap_{i\in\mathcal{P}} \hat{H}_i$
$\Gamma(M)$	множество граничных точек допустимого многогранника M
$\mathcal{I}$	множество индексов, для которых полупространство $\hat{H}_i$ является рецессивным
$\hat{M}$	рецессивный многогранник, определяемый формулой $\hat{M} = \bigcap_{i\in\mathcal{I}} \hat{H}_i$
$\Gamma(\hat{M})$	множество граничных точек рецессивного многогранника $\hat{M}$
$\boldsymbol{\gamma}_i(\boldsymbol{z})$	целевая проекция точки $\boldsymbol{z}$ на гиперплоскость H_i
$\beta_i(\boldsymbol{z})$	целевое смещение точки $\boldsymbol{z}$ относительно гиперплоскости H_i: $\|\beta_i(\boldsymbol{z})\| = \|\boldsymbol{\gamma}_i(\boldsymbol{z}) - \boldsymbol{z}\|$
$\hat{\boldsymbol{\gamma}}(\boldsymbol{z})$	целевая проекция точки $\boldsymbol{z}$ на границу $\Gamma(\hat{M})$ рецессивного многогранника $\hat{M}$
$\hat{\beta}(\boldsymbol{z})$	целевое смещение точки $\boldsymbol{z}$ относительно границы $\Gamma(\hat{M})$: $\|\hat{\beta}(\boldsymbol{z})\| = \|\hat{\boldsymbol{\gamma}}(\boldsymbol{z}) - \boldsymbol{z}\|$
$V_r(\boldsymbol{x})$	гипершар радиуса r с центром в точке $\boldsymbol{x}$

Список литературы

1. *Hartung T.* Making Big Sense from Big Data // Frontiers in Big Data. 2018. Vol. 1. P. 5. doi 10.3389/fdata.2018.00005.
2. *Соколинская И.М., Соколинский Л.Б.* О решении задачи линейного программирования в эпоху больших данных // Параллельные вычислительные технологии (ПаВТ'2017). Короткие статьи и описания плакатов. Челябинск: Издательский центрЮУрГУ, 2017. С. 471–484. http://omega.sp.susu.ru/pavt2017/short/014.pdf.
3. *Branke J.* Optimization in Dynamic Environments // Evolutionary Optimization in Dynamic Environments. Genetic Algorithms and Evolutionary Computation, vol. 3. Boston, MA: Springer, 2002. P. 13–29. doi 10.1007/978-1-4615-0911-0_2.
4. *Ерёмин И.И., Мазуров В.Д.* Нестационарные процессы математического программирования. М.: Наука. Главная редакция физико-математической литературы, 1979. 288 с.

5. *Brogaard J., Hendershott T., Riordan R.* High-Frequency Trading and Price Discovery // Review of Financial Studies. 2014. Vol. 27, no. 8. P. 2267–2306. doi 10.1093/rfs/hhu032.
6. *Deng S., Huang X., Wang J., et al.* A Decision Support System for Trading in Apple Futures Market Using Predictions Fusion // IEEE Access. 2021. Vol. 9. P. 1271–1285. doi 10.1109/ACCESS.2020.3047138.
7. *Seregin G.* Lecture notes on regularity theory for the Navier-Stokes equations. Singapore: World Scientific Publishing Company, 2014. 268 p. doi 10.1142/9314.
8. *Demin D.A.* Synthesis of optimal control of technological processes based on a multialternative parametric description of the final state // Eastern-European Journal of Enterprise Technologies. 2017. Vol. 3, 4(87). P. 51–63. doi 10.15587/1729-4061.2017.105294.
9. *Kazarinov L.S., Shnayder D.A., Kolesnikova O.V.* Heat load control in steam boilers // 2017 International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2017 - Proceedings. IEEE, 2017. doi 10.1109/ICIEAM.2017.8076177.
10. *Zagoskina E.V., Barbasova T.A., Shnaider D.A.* Intelligent Control System of Blast-furnace Melting Ffiency // SIBIRCON 2019 - International Multi-Conference on Engineering, Computer and Information Sciences, Proceedings. IEEE, 2019. P. 710–713. doi 10.1109/SIBIRCON48586.2019.8958221.
11. *Fleming J., Yan X., Allison C., et al.* Real-time predictive eco-driving assistance considering road geometry and long-range radar measurements // IET Intelligent Transport Systems. 2021. Vol. 15, no. 4. P. 573–583. doi 10.1049/ITR2.12047.
12. *Scholl M., Minnerup K., Reiter C., et al.* Optimization of a thermal management system for battery electric vehicles // 14th International Conference on Ecological Vehicles and Renewable Energies, EVER 2019. IEEE, 2019. doi 10.1109/EVER.2019.8813657.
13. *Meisel S.* Dynamic Vehicle Routing // Anticipatory Optimization for Dynamic Decision Making. Operations Research/Computer Science Interfaces Series, vol. 51. New York, NY: Springer, 2011. P. 77–96. doi 10.1007/978-1-4614-0505-4_6.
14. *Kiran D* Production Planning and Control: A Comprehensive Approach. Elsevier Inc., 2019. 582 p. doi 10.1016/C2018-0-03856-6.
15. *Mall R.* Real-Time Systems: Theory and Practice. Delhi, India: Pearson Education, 2007. 242 p.
16. *Dantzig G.B.* Linear programming and extensions. Princeton, N.J.: Princeton university press, 1998. 656 p.
17. *Hall J., McKinnon K.* Hyper-sparsity in the revised simplex method and how to exploit it // Computational Optimization and Applications. 2005. Vol. 32, no. 3. P. 259–283. doi 10.1007/s10589-005-4802-0.
18. *Klee V., Minty G.* How good is the simplex algorithm? // Inequalities - III. Proceedings of the Third Symposium on Inequalities Held at the University of California, Los Angeles, Sept. 1-9, 1969 / ed. by O. Shisha. New York-London: Academic Press, 1972. P. 159–175.
19. *Bartels R., Stoer J., Zenger C.* A Realization of the Simplex Method Based on Triangular Decompositions // Handbook for Automatic Computation. Volume II: Linear Algebra. Berlin, Heidelberg: Springer, 1971. P. 152–190. doi 10.1007/978-3-642-86940-2_11.
20. *Tolla P.* A Survey of Some Linear Programming Methods // Concepts of Combinatorial Optimization / ed. by V.T. Paschos. 2nd ed. Hoboken, NJ, USA: John Wiley, Sons, 2014. Chap. 7. P. 157–188. doi 10.1002/9781119005216.ch7.
21. *Hall J.* Towards a practical parallelisation of the simplex method // Computational Management Science. 2010. Vol. 7, no. 2. P. 139–170. doi 10.1007/s10287-008-0080-5.
22. *Mamalis B., Pantziou G.* Advances in the Parallelization of the Simplex Method // Algorithms, Probability, Networks, and Games. Lecture Notes in Computer Science, vol. 9295 / ed. by C. Zaroliagis, G. Pantziou, S. Kontogiannis. Cham: Springer, 2015. P. 281–307. doi 10.1007/978-3-319-24024-4_17.
23. *Зоркальцев В.И., Мокрый И.В.* Алгоритмы внутренних точек в линейной оптимизации // Сибирский журнал индустриальной математики. 2018. Т. 21, 1 (73). С. 11–20. doi 10.17377/sibjim.2018.21.102.
24. *Дикин И.И.* Итеративное решение задач линейного и квадратичного программирования // Доклады Академии наук СССР. 1967. Т. 174, № 4. С. 747–748. https://www.mathnet.ru/rus/dan33112.
25. *Gondzio J.* Interior point methods 25 years later // European Journal of Operational Research. 2012. Vol. 218, no. 3. P. 587–601. doi 10.1016/j.ejor.2011.09.017.
26. *Roos C., Terlaky T., Vial J.-P.* Interior Point Methods for Linear Optimization. New York: Springer, 2005. 500 p. doi 10.1007/b100325.
27. *Sokolinskaya I.M.* Parallel Method of Pseudoprojection for Linear Inequalities // Parallel Computational Technologies. PCT 2018. Communications in Computer and Information Science, vol. 910 / ed. by L. Sokolinsky, M. Zymbler. Cham: Springer, 2018. P. 216–231. doi 10.1007/978-3-319-99673-8_16.

28. *Gondzio J., Grothey A.* Direct Solution of Linear Systems of Size 109 Arising in Optimization with Interior Point Methods // Parallel Processing and Applied Mathematics. PPAM 2005. Lecture Notes in Computer Science, vol. 3911. 3911 LNCS / ed. by R. Wyrzykowski, J. Dongarra, N. Meyer, J. Wasniewski. Berlin, Heidelberg: Springer, 2006. P. 513–525. doi 10.1007/11752578_62.
29. *Prieto A., Prieto B., Ortigosa E.M., et al.* Neural networks: An overview of early research, current frameworks and new challenges // Neurocomputing. 2016. Vol. 214. P. 242–268. doi 10.1016/j.neucom.2016.06.014.
30. *Tank D.W., Hopfield J.J.* Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit // IEEE transactions on circuits and systems. 1986. Vol. CAS–33, no. 5. P. 533–541. doi 10.1109/TCS.1986.1085953.
31. *Kennedy M.P., Chua L.O.* Unifying the Tank and Hopfield Linear Programming Circuit and the Canonical Nonlinear Programming Circuit of Chua and Lin // IEEE Transactions on Circuits and Systems. 1987. Vol. 34, no. 2. P. 210–214. doi 10.1109/TCS.1987.1086095.
32. *Rodriguez-Vazquez A., Dominguez-Castro R., Rueda A., et al.* Nonlinear Switched-Capacitor "Neural" Networks for Optimization Problems // IEEE Transactions on Circuits and Systems. 1990. Vol. 37, no. 3. P. 384–398. doi 10.1109/31.52732.
33. *Zak S.H., Upatising V.* Solving Linear Programming Problems with Neural Networks: A Comparative Study // IEEE Transactions on Neural Networks. 1995. Vol. 6, no. 1. P. 94–104. doi 10.1109/72.363446.
34. *Malek A., Yari A.* Primal-dual solution for the linear programming problems using neural networks // Applied Mathematics and Computation. 2005. Vol. 167, no. 1. P. 198–211. doi 10.1016/J.AMC.2004.06.081.
35. *Liu X., Zhou M.* A one-layer recurrent neural network for non-smooth convex optimization subject to linear inequality constraints // Chaos, Solitons and Fractals. 2016. Vol. 87. P. 39–46. doi 10.1016/j.chaos.2016.03.009.
36. *LeCun Y., Bengio Y., Hinton G.* Deep learning // Nature. 2015. Vol. 521, no. 7553. P. 436–444. doi 10.1038/nature14539.
37. *Ольховский Н.А., Соколинский Л.Б.* Визуальное представление многомерных задач линейного программирования // ВестникЮУрГУ. Серия: Вычислительная математика и информатика. 2022. Т. 11, № 1. С. 31–56. doi 10.14529/cmse220103.
38. *Raina R., Madhavan A., Ng A.Y.* Large-scale deep unsupervised learning using graphics processors // Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09). New York, NY, USA: ACM Press, 2009. P. 873–880. doi 10.1145/1553374.1553486.
39. *Соколинский Л.Б., Соколинская И.М.* О новой версии апекс-метода для решения задач линейного программирования // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2023. Т. 12, № 2. С. 5–46. doi 10.14529/cmse230201.
40. *Ольховский Н.А.* Исследование структуры рецептивного поля в визуальном методе решения задачи линейного программирования. PREPRINTS.RU. doi 10.24108/preprints-3112771.

Поступила в редакцию
16 июня 2021 г.

Принята к публикации
13 октября 2021 г.

Информация об авторах

Николай Александрович Ольховский — аспирант; Южно-Уральский государственный университет (национальный исследовательский университет), пр. им. В.И. Ленина, д. 76, 454080, Челябинск, Российская Федерация.

Леонид Борисович Соколинский — д.ф.-м.н., зав. каф. системного программирования; Южно-Уральский государственный университет (национальный исследовательский университет), пр. им. В.И. Ленина, д. 76, 454080, Челябинск, Российская Федерация.

References

1. *Hartung T.* Making Big Sense From Big Data. Frontiers in Big Data. 2018. Vol. 1. P. 5. doi 10.3389/fdata.2018.00005.
2. *Sokolinskaya I.M., Sokolinsky L.B.* On the Solution of Linear Programming Problems in the Age of Big Data. Parallel Computational Technologies. PCT 2017. Communications in Computer and Information Science, vol. 753. / ed. by L. Sokolinsky, M. Zymbler. Cham: Springer, 2017. P. 86–100. doi 10.1007/978-3-319-67035-5_7.
3. *Branke J.* Optimization in Dynamic Environments. Evolutionary Optimization in Dynamic Environments. Genetic Algorithms and Evolutionary Computation, vol. 3. Boston, MA: Springer, 2002. P. 13–29. doi 10.1007/978-1-4615-0911-0_2.
4. *Eremin I.I., Mazurov V.D.* Nonstationary processes of mathematical programming (Nestacionarnye processy matematicheskogo programmirovaniya). Moscow: Nauka. Glavnaya redakciya fiziko-matematicheskoj literatury, 1979. 288 p. (in Russian).
5. *Brogaard J., Hendershott T., Riordan R.* High-Frequency Trading and Price Discovery. Review of Financial Studies. 2014. Vol. 27, no. 8. P. 2267–2306. doi 10.1093/rfs/hhu032.
6. *Deng S., Huang X., Wang J., et al.* A Decision Support System for Trading in Apple Futures Market Using Predictions Fusion. IEEE Access. 2021. Vol. 9. P. 1271–1285. doi 10.1109/ACCESS.2020.3047138.
7. *Seregin G.* Lecture notes on regularity theory for the Navier-Stokes equations. Singapore: World Scientific Publishing Company, 2014. 268 p. doi 10.1142/9314.
8. *Demin D.A.* Synthesis of optimal control of technological processes based on a multialternative parametric description of the final state. Eastern-European Journal of Enterprise Technologies. 2017. Vol. 3, 4(87). P. 51–63. doi 10.15587/1729-4061.2017.105294.
9. *Kazarinov L.S., Shnayder D.A., Kolesnikova O.V.* Heat load control in steam boilers. 2017 International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2017 - Proceedings. IEEE, 2017. doi 10.1109/ICIEAM.2017.8076177.
10. *Zagoskina E.V., Barbasova T.A., Shnaider D.A.* Intelligent Control System of Blast-furnace Melting Ffiency. SIBIRCON 2019 - International Multi-Conference on Engineering, Computer and Information Sciences, Proceedings. IEEE, 2019. P. 710–713. doi 10.1109/SIBIRCON48586.2019.8958221.
11. *Fleming J., Yan X., Allison C., et al.* Real-time predictive eco-driving assistance considering road geometry and long-range radar measurements. IET Intelligent Transport Systems. 2021. Vol. 15, no. 4. P. 573–583. doi 10.1049/ITR2.12047.
12. *Scholl M., Minnerup K., Reiter C., et al.* Optimization of a thermal management system for battery electric vehicles. 14th International Conference on Ecological Vehicles and Renewable Energies, EVER 2019. IEEE, 2019. doi 10.1109/EVER.2019.8813657.
13. *Meisel S.* Dynamic Vehicle Routing. Anticipatory Optimization for Dynamic Decision Making. Operations Research/Computer Science Interfaces Series, vol. 51. New York, NY: Springer, 2011. P. 77–96. doi 10.1007/978-1-4614-0505-4_6.
14. *Kiran D* Production Planning and Control: A Comprehensive Approach. Elsevier Inc., 2019. 582 p. doi 10.1016/C2018-0-03856-6.
15. *Mall R.* Real-Time Systems: Theory and Practice. Delhi, India: Pearson Education, 2007. 242 p.
16. *Dantzig G.B.* Linear programming and extensions. Princeton, N.J.: Princeton university press, 1998. 656 p.
17. *Hall J., McKinnon K.* Hyper-sparsity in the revised simplex method and how to exploit it. Computational Optimization and Applications. 2005. Vol. 32, no. 3. P. 259–283. doi 10.1007/s10589-005-4802-0.
18. *Klee V., Minty G.* How good is the simplex algorithm? Inequalities - III. Proceedings of the Third Symposium on Inequalities Held at the University of California, Los Angeles, Sept. 1-9, 1969 / ed. by O. Shisha. New York-London: Academic Press, 1972. P. 159–175.
19. *Bartels R., Stoer J., Zenger C.* A Realization of the Simplex Method Based on Triangular Decompositions. Handbook for Automatic Computation. Volume II: Linear Algebra. Berlin, Heidelberg: Springer, 1971. P. 152–190. doi 10.1007/978-3-642-86940-2_11.
20. *Tolla P.* A Survey of Some Linear Programming Methods. Concepts of Combinatorial Optimization / ed. by V.T. Paschos. 2nd ed. Hoboken, NJ, USA: John Wiley, Sons, 2014. Chap. 7. P. 157–188. doi 10.1002/9781119005216.ch7.
21. *Hall J.* Towards a practical parallelisation of the simplex method. Computational Management Science. 2010. Vol. 7, no. 2. P. 139–170. doi 10.1007/s10287-008-0080-5.

22. *Mamalis B., Pantziou G.* Advances in the Parallelization of the Simplex Method. Algorithms, Probability, Networks, and Games. Lecture Notes in Computer Science, vol. 9295 / ed. by C. Zaroliagis, G. Pantziou, S. Kontogiannis. Cham: Springer, 2015. P. 281–307. doi 10.1007/978-3-319-24024-4_17.
23. *Zorkaltsev V., Mokryi I.* Interior point algorithms in linear optimization. Journal of applied and industrial mathematics. 2018. Vol. 12, no. 1. P. 191–199. doi 10.1134/S1990478918010179.
24. *Dikin I.* Iterative solution of problems of linear and quadratic programming. Soviet Mathematics. Doklady. 1967. Vol. 8. P. 674–675.
25. *Gondzio J.* Interior point methods 25 years later. European Journal of Operational Research. 2012. Vol. 218, no. 3. P. 587–601. doi 10.1016/j.ejor.2011.09.017.
26. *Roos C., Terlaky T., Vial J.-P.* Interior Point Methods for Linear Optimization. New York: Springer, 2005. 500 p. doi 10.1007/b100325.
27. *Sokolinskaya I.* Parallel Method of Pseudoprojection for Linear Inequalities. Parallel Computational Technologies. PCT 2018. Communications in Computer and Information Science, vol. 910 / ed. by L. Sokolinsky, M. Zymbler. Cham: Springer, 2018. P. 216–231. doi 10.1007/978-3-319-99673-8_16.
28. *Gondzio J., Grothey A.* Direct Solution of Linear Systems of Size 109 Arising in Optimization with Interior Point Methods. Parallel Processing and Applied Mathematics. PPAM 2005. Lecture Notes in Computer Science, vol. 3911. 3911 LNCS / ed. by R. Wyrzykowski, J. Dongarra, N. Meyer, J. Wasniewski. Berlin, Heidelberg: Springer, 2006. P. 513–525. doi 10.1007/11752578_62.
29. *Prieto A., Prieto B., Ortigosa E.M., et al.* Neural networks: An overview of early research, current frameworks and new challenges. Neurocomputing. 2016. Vol. 214. P. 242–268. doi 10.1016/j.neucom.2016.06.014.
30. *Tank D.W., Hopfield J.J.* Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit. IEEE transactions on circuits and systems. 1986. Vol. CAS–33, no. 5. P. 533–541. doi 10.1109/TCS.1986.1085953.
31. *Kennedy M.P., Chua L.O.* Unifying the Tank and Hopfield Linear Programming Circuit and the Canonical Nonlinear Programming Circuit of Chua and Lin. IEEE Transactions on Circuits and Systems. 1987. Vol. 34, no. 2. P. 210–214. doi 10.1109/TCS.1987.1086095.
32. *Rodriguez-Vazquez A., Dominguez-Castro R., Rueda A., et al.* Nonlinear Switched-Capacitor "Neural" Networks for Optimization Problems. IEEE Transactions on Circuits and Systems. 1990. Vol. 37, no. 3. P. 384–398. doi 10.1109/31.52732.
33. *Zak S.H., Upatising V.* Solving Linear Programming Problems with Neural Networks: A Comparative Study. IEEE Transactions on Neural Networks. 1995. Vol. 6, no. 1. P. 94–104. doi 10.1109/72.363446.
34. *Malek A., Yari A.* Primal-dual solution for the linear programming problems using neural networks. Applied Mathematics and Computation. 2005. Vol. 167, no. 1. P. 198–211. doi 10.1016/J.AMC.2004.06.081.
35. *Liu X., Zhou M.* A one-layer recurrent neural network for non-smooth convex optimization subject to linear inequality constraints. Chaos, Solitons and Fractals. 2016. Vol. 87. P. 39–46. doi 10.1016/j.chaos.2016.03.009.
36. *LeCun Y., Bengio Y., Hinton G.* Deep learning. Nature. 2015. Vol. 521, no. 7553. P. 436–444. doi 10.1038/nature14539.
37. *Olkhovsky N., Sokolinsky L.* Visualizing Multidimensional Linear Programming Problems. Parallel Computational Technologies. PCT 2022. Communications in Computer and Information Science, vol. 1618 / ed. by L. Sokolinsky, M. Zymbler. Cham: Springer, 2022. P. 172–196. doi 10.1007/978-3-031-11623-0_13.
38. *Raina R., Madhavan A., Ng A.Y.* Large-scale deep unsupervised learning using graphics processors. Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09). New York, NY, USA: ACM Press, 2009. P. 873–880. doi 10.1145/1553374.1553486.
39. *Sokolinsky L.B., Sokolinskaya I.M.* Apex Method: A New Scalable Iterative Method for Linear Programming. Mathematics. 2023. Vol. 11, no. 7. P. 1654. doi 10.3390/MATH11071654.
40. *Olkhovsky N.A.* Study of the receptive field structure in the visual method of solving the linear programming problem. PREPRINTS.RU. doi 10.24108/preprints-3112771. (In Russian)

Received
June 16, 2021

Accepted for publication
October 13, 2021

Information about the authors

Nikolay A. Olkhovsky — PhD student; South Ural State University (National Research University), pr. Lenina 76, 454080, Chelyabinsk, Russia.

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

Leoind B. Sokolinsky — Doctor of sciences (phys. and math.), Head of Comp. Sc. Dept.; South Ural State University (National Research University), pr. Lenina 76, 454080, Chelyabinsk, Russia.

Главному редактору научного журнала

“Вычислительные методы и программирование”

чл.-корр. РАН, проф. Вл. В. Воеводину

СОПРОВОДИТЕЛЬНОЕ ПИСЬМО

Направляем в журнал “Вычислительные методы и программирование” пересмотренную версию статьи “О новом методе линейного программирования”. В данной версии учтены все замечания рецензента. Ответы на замечания прилагаем отдельным документом.

Сообщаем следующую информацию о каждом авторе статьи.

1. Фамилия, имя, отчество полностью: Ольховский Николай Александрович (Olkhovsky Nikolay A.).

2. Основное место работы: Южно-Уральский государственный университет (национальный исследовательский университет) (South Ural State University (national research university)) http://www.susu.ru/.

3. Должность по основному месту работы: аспирант (PhD student).

4. Ученая степень и ученое звание: нет (no).

5. Телефон мобильный: +7 9043070939.

6. E-mail: olkhovskiina@susu.ru.

7. ORCID: 0009-0008-9078-4799.

8. Номер и название рубрики ГРНТИ: 27.47.19 исследование операций; код OECD: 1.02.

9. Ссылка на профиль автора в elibrary.ru: https://elibrary.ru/author_items.asp?authorid=1200111.

1. Фамилия, имя, отчество полностью: Соколинский Леонид Борисович (Sokolinsky Leonid B.).

2. Основное место работы: Южно-Уральский государственный университет (национальный исследовательский университет) (South Ural State University (national research university)) http://www.susu.ru/.

3. Должность по основному месту работы: заведующий кафедрой системного программирования (Head of Computer Science Department).

4. Ученая степень и ученое звание: доктор физико-математических наук, профессор (Doctor of Sciences, Professor).

5. Телефон мобильный: +7 9128912525.

6. E-mail: leonid.sokolinsky@susu.ru.

7. ORCID: 0000-0001-9997-3918.

8. Номер и название рубрики ГРНТИ: 27.47.19 исследование операций; код OECD: 1.02.

9. Ссылка на профиль автора в elibrary.ru: https://elibrary.ru/author_items.asp

2021, ?? (?), 1–26. doi 10.26089/NumMet.v??r???

?authorid=99314.

Текущую переписку по вопросам публикации статьи следует вести с Соколинским Л.Б.

Авторы согласны с правилами подготовки статьи к публикации, включая рецензирование, научное и литературное редактирование и доведение статьи до редакторских стандартов, принятых в рамках журнала.

Авторы ознакомлены с Публикационной этикой журнала и согласны с ее положениями.

Авторы также согласны с передачей журналу своего права на издание и распространение статьи в электронной и бумажной версиях, в том числе на размещение библиографической информации о статье в Российском индексе научного цитирования (РИНЦ) и в других базах научного цитирования и на размещение полных текстов статей в Научной электронной библиотеке (elibrary.ru) и портале http://www.mathnet.ru для свободного доступа всем пользователям Интернет независимо от их категории и местоположения.

23.05.2023 _____ Н.А. Ольховский

_____ Л.Б. Соколинский