АЛГОРИТМЫ СКОШЕННОГО ПУТИ ДЛЯ РЕШЕНИЯ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ И СИСТЕМ ЛИНЕЙНЫХ НЕРАВЕНСТВ [1]

Филатов Александр Юрьевич

(Институт систем энергетики им. Л.А. Мелентьева, e-mail: fial@isem.sei.irk.ru)

Обобщаются полученные автором результаты по разработке, теоретическому обоснованию и экспериментальному исследованию нескольких вариантов алгоритмов скошенного пути для решения задач линейного программирования и систем линейных неравенств. Предложенные алгоритмы развивают идеи алгоритмов центрального пути, в то же время вычислительный процесс в них может начинаться с любых относительно внутренних точек множеств допустимых решений, что существенно ослабляет проблему инициализации.

ВВЕДЕНИЕ

При решении многих задач из различных прикладных областей широкое применяются методы линейной оптимизации. В частности, к задачам линейного программирования и тесно связанным с ними системам линейных неравенств сводятся многие существенно нелинейные модели, при реализации которых используется итеративная линеаризация.

Существует большое количество методов решения задач линейного программирования. Одним из активно развивающихся их направлений являются алгоритмы внутренних точек, название которых связано с тем, что, в отличие от симплекс-метода, перебирающего угловые точки многогранника допустимых решений, вычислительный процесс в них происходит в относительной внутренности допустимого множества. Идейные основы алгоритмов внутренних точек были заложены в 50–60-х годах XX века, в частности в работах К.Фриша [1], А.Фиакко и Г.Мак-Кормика [2].

[1] Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект №00-01-00530).

Повышенный интерес к данному направлению (результатом которого стали более 2000 опубликованных статей) в мировой науке возник вслед за созданием в 1984 г. Н. Кармаркаром [3] первого алгоритма внутренних точек, обладающего полиномиальными от размерности задачи оценками максимального объема вычислений, необходимых для ее решения. Следует отметить, что формулы алгоритма Кармаркара очень близки к формулам алгоритма, опубликованного в 1967 г. И.И. Дикиным [4].

Алгоритмы с гарантированными теоретическими оценками объема вычислений могут оказаться особенно важными для решения задач управления объектами в темпе реального времени, когда заранее необходимо знать максимальное время получения решения с требуемой точностью. В то же время в полиномиальных алгоритмах оценка строится на основе наихудшего возможного случая, вероятность реализации которого на практике невелика. Известно, что многие полиномиальные алгоритмы малопригодны для решения практических задач, т. е. наличие полиномиальных оценок максимального объема вычислений не означает хорошей работы алгоритмов на практике. Поэтому актуальна разработка специальных процедур, позволяющих ускорить вычислительный процесс и сделать полиномиальные алгоритмы более привлекательными для практического использования.

Одной из серьезных проблем, требующих решения, является инициализация алгоритма. Это вызвано тем, что для многих полиномиальных алгоритмов необходимо стартовое приближение, обладающее особыми свойствами, а процедуры его автоматического формирования базируются на идее сведения исходной задачи к расширенной, малопривлекательной с вычислительной точки зрения задаче специального вида.

Алгоритмы внутренних точек допускают множество вариантов в зависимости от вида решаемой задачи. В частности, мы уделим особое внимание проблеме решения алгоритмами внутренних точек систем линейных неравенств с двухсторонними ограничениями на переменные. Подобные интервальные ограничения могут быть вызваны как технически возможными пределами изменения параметров системы, так и искусственными ограничениями на ряд показателей. Проблема решения систем линейных уравнений и неравенств с двухсторонними ограничениями на переменные

часто возникает при применении методов последовательной линеаризации к нелинейным моделям. В то же время, до сих пор в алгоритмах слабо учитывались полезные свойства таких систем, позволяющие, в частности, решить проблему быстрой идентификации несовместности ограничений.

Рассмотрим пару взаимно-двойственных задач линейного программирования:

$$\mathbf{c}^{\mathbf{T}}\mathbf{x} \to \min_{\mathbf{x}\in\mathbf{X}},\ \mathbf{X} = \left\{\mathbf{x} \in \mathbf{R}^{\mathbf{n}}:\ \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\right\}, \tag{0.1}$$

$$\mathbf{b}^{\mathbf{T}}\mathbf{u} \to \max_{\mathbf{u}\in\mathbf{U}},\ \mathbf{U} = \left\{\mathbf{u} \in \mathbf{R}^{\mathbf{m}}:\ \mathbf{g}(\mathbf{u}) \equiv \mathbf{c} - \mathbf{A}^{\mathbf{T}}\mathbf{u} \geq \mathbf{0}\right\}, \tag{0.2}$$

где $\mathbf{c} \in \mathbf{R}^{\mathbf{n}}$, $\mathbf{b} \in \mathbf{R}^{\mathbf{m}}$, $\mathbf{A}$ – матрица размерности $m \times n$, $rank\,\mathbf{A} = m$.

Пара задач (0.1)–(0.2) равносильна следующей самосопряженной задаче:

$$\mathbf{c}^{\mathbf{T}}\mathbf{x} - \mathbf{b}^{\mathbf{T}}\mathbf{u} \to \min_{\mathbf{x}\in\mathbf{X},\mathbf{u}\in\mathbf{U}}.$$

Поскольку важным является практическое использование разработанных алгоритмов, необходимо сравнить их по быстродействию с некоторым эталоном. В качестве “эталонных” могут выступать аффинно-масштабирующие алгоритмы, не обладающие полиномиальными оценками, но хорошо зарекомендовавшие себя на практике. Вычислительный процесс в них начинается с любого стартового приближения $\mathbf{x}^{\mathbf{1}} > \mathbf{0}$. На каждой итерации ищется вектор невязок балансовых ограничений-равенств

$$\mathbf{r}^{\mathbf{k}} = \mathbf{b} - \mathbf{A}\mathbf{x}^{\mathbf{k}}.$$

Итеративный переход осуществляется по правилу

$$\mathbf{x}^{\mathbf{k+1}} = \mathbf{x}^{\mathbf{k}} + \lambda^k \Delta\mathbf{x}^{\mathbf{k}},\quad k = 1,2,\ldots,$$

где $\Delta\mathbf{x}^{\mathbf{k}}$ – направление корректировки, вычисляемое как решение задачи

$$\mathbf{c}^{\mathbf{T}}\Delta\mathbf{x} + \frac{1}{2}\sum_{j=1}^{n} \Delta x_j^2 \big/ d_j^k \to \min_{\Delta\mathbf{x}\in\mathbf{R}^{\mathbf{n}}},\quad \mathbf{A}\Delta\mathbf{x} = \mathbf{r}^{\mathbf{k}},$$

а λ^k – шаг корректировки, задаваемый из соображений невыхода переменных прямой задачи за пределы внутренности допустимой области:

$$\lambda^k = \gamma \min_{j:\Delta x_j^k < 0}\left(-x_j^k \big/ \Delta x_j^k\right).$$

Здесь γ – заданная константа из интервала (0;1). Отметим, что на фазе ввода в допустимую область (т. е. при $\mathbf{r}^{\mathbf{k}} \neq \mathbf{0}$) шаг λ^k должен ограничиваться

сверху единицей.

Среди способов выбора весовых коэффициентов d_j^k выделим следующие:

$$1)\ d_j^k = \left(x_j^k\right)^2,\ \ j = 1,\ldots,n; \tag{0.3}$$

$$2)\ d_j^k = x_j^k \Big/ \max\left\{\varepsilon, g_j\left(\mathbf{u}^{\mathbf{k-1}}\right)\right\},\ \ j = 1,\ldots,n,\ \ \varepsilon > 0. \tag{0.4}$$

Правило (0.4) было введено [5] в целях уменьшения негативного влияния погрешностей вычислений, особенно в финальной стадии вычислительного процесса, и увеличения численной устойчивости алгоритма. Существует множество других правил выбора весов, но в качестве "эталонных" возьмем именно эти два варианта.

Предлагаемые алгоритмы скошенного пути являются развитием и обобщением алгоритмов центрального пути, которые строятся на основе идеи К. Фриша [1] включения в целевую функцию штрафных слагаемых в виде логарифма ограничений-неравенств с параметром, монотонно уменьшающимся до нуля. Точным решением задачи минимизации логарифмической барьерной функции на множествах допустимых решений пары задач (0.1)–(0.2)

$$f_\mu(\mathbf{x},\mathbf{u}) = \mathbf{c}^{\mathbf{T}}\mathbf{x} - \mathbf{b}^{\mathbf{T}}\mathbf{u} - \mu\sum_{j=1}^{n}\ln\left(x_j g_j(\mathbf{u})\right) \to \min_{\mathbf{x}\in\mathbf{X},\mathbf{u}\in\mathbf{U}}$$

при любом $\mu > 0$ является пара векторов $\mathbf{x}(\mu), \mathbf{u}(\mu)$, для которой все компоненты вектора невязки двойственности равны между собой:

$$x_j(\mu) g_j\left(\mathbf{u}(\mu)\right) = \mu,\ \ j = 1,\ldots,n.$$

Множество пар векторов $\mathbf{x}(\mu), \mathbf{u}(\mu)$ при всех $\mu > 0$ образует центральный путь.

Суть алгоритмов центрального пути состоит в том, что вырабатываются последовательности пар векторов $\left(\mathbf{x}^{\mathbf{k}}, \mathbf{u}^{\mathbf{k}}\right)$ и соответствующих значений μ^k, для которых при заданном $\theta \in (0;1)$ справедливы условия

$$\mathbf{x}^{\mathbf{k}} \in \mathbf{X},\ \ \mathbf{u}^{\mathbf{k}} \in \mathbf{U},\ \ \sum_{j=1}^{n}\frac{1}{\mu^k}\left(\mu^k - x_j^k g_j\left(\mathbf{u}^{\mathbf{k}}\right)\right)^2 \le \theta\mu^k. \tag{0.5}$$

При этом для некоторого $\beta > 0$ выполняется неравенство

$$\mu^{k+1} \le \left(1 - \beta/\sqrt{n}\right)\mu^k,$$

что обеспечивает [6] получение решения пары задач (0.1)–(0.2) за $O\left(\sqrt{n}L\right)$ итераций. Здесь и далее L – объем [6] входных данных задачи.

Классическим для данного класса является алгоритм М. Коджимы и др. [7], в котором значение β может равняться 0.125, и алгоритм, разработанный Р.Монтейро и И.Адлером [6], где $\beta = 0.35$. В работах [8], [9] получено обоснование для $\beta = 0.5$, что является на сегодняшний день рекордным значением. Рассматриваемые ниже алгоритмы скошенного пути являются развитием и обобщением вариантов алгоритмов из [8] и [9].

В отличие от аффинно-масштабирующих алгоритмов для алгоритмов центрального пути требуется начальное приближение, удовлетворяющее при $k = 1$ условиям (0.5). Существующие процедуры [6], [10] его автоматического формирования базируются на идее сведения исходной задачи к расширенной, малопривлекательной с вычислительной точки зрения, задаче специального вида. В связи с этим важной целью является разработка алгоритмов, не предъявляющих жестких требований к стартовому приближению.

1. АЛГОРИТМЫ СКОШЕННОГО ПУТИ

Таким свойством обладает предложенный и обоснованный в [11] класс алгоритмов оптимизации в конусе “скошенного пути”. Ключевой особенностью этих алгоритмов является то, что вычислительный процесс в них может начинаться с любых относительно внутренних точек множеств допустимых решений задач (0.1)–(0.2). Идейно алгоритмы скошенного пути наиболее близки к введенным Б. Янсеном и др. [12] методам “следования цели”.

Предположение 1. *Будем считать, что задачи* (0.1)–(0.2) *имеют допустимые векторы, для которых все ограничения-неравенства выполняются в строгой форме, т. е. имеются такие векторы* $\mathbf{x} \in \mathbf{X}, \mathbf{u} \in \mathbf{U}$*, что*

$$x_j > 0,\ \ g_j(\mathbf{u}) > 0,\ \ j = 1,\ldots,n.$$

Теорема 1 [13]. *Пусть для задач* (0.1)–(0.2) *выполняется предположение* 1. *Тогда для любого заданного вектора* $\mathbf{t} > \mathbf{0}$ *существует и единственна такая пара векторов* $\mathbf{x}(\mathbf{t}), \mathbf{u}(\mathbf{t})$*, что выполняются условия*

$$\mathbf{x}(\mathbf{t}) \in \mathbf{X},\ \mathbf{u}(\mathbf{t}) \in \mathbf{U},\ \ x_j(\mathbf{t}) g_j(\mathbf{u}(\mathbf{t})) = t_j,\ \ j = 1,\ldots,n.$$

Скошенным путем, инициируемым вектором $\mathbf{t} > \mathbf{0}$, назовем множество пар векторов $\mathbf{x}(\mu\mathbf{t}), \mathbf{u}(\mu\mathbf{t})$ при всех $\mu > 0$. В частности, вектор $\mathbf{t} = \mathbf{e}$ (равно как и $\mathbf{t} = \lambda\mathbf{e}$ при любом $\lambda > 0$) инициирует центральный путь.

Каждый скошенный путь характеризуется коэффициентом скошенности

$$\gamma = \bar{t}/t_{\min},$$

где $\bar{t} = \frac{1}{n}\sum_{j=1}^{n} t_j,\ t_{\min} = \min\limits_{j=1,\ldots,n} t_j$.

Поскольку для пути аналитических центров все компоненты инициирующего вектора равны между собой, его коэффициент скошенности равен единице. Для всех остальных скошенных путей он больше единицы.

Все предлагаемые алгоритмы вырабатывают последовательности пар векторов $\left(\mathbf{x}^{\mathbf{k}}, \mathbf{u}^{\mathbf{k}}\right)$ и соответствующих значений μ^k, для которых при заданном $\theta \in (0;1)$ выполняются условия принадлежности конусу скошенного пути:

$$\mathbf{x}^{\mathbf{k}} \in \mathbf{X},\ \mathbf{u}^{\mathbf{k}} \in \mathbf{U},$$
$$\Phi_2\left(\mathbf{x}^{\mathbf{k}}, \mathbf{u}^{\mathbf{k}}, \mu^k\right) \equiv \sum_{j=1}^{n} \frac{1}{\mu^k t_j}\left(\mu^k t_j - x_j^k g_j\left(\mathbf{u}^{\mathbf{k}}\right)\right)^2 \le \theta \mu^k t_{\min}. \qquad (1.1)$$

Схематически траектория алгоритмов скошенного пути показана на рис. 1.

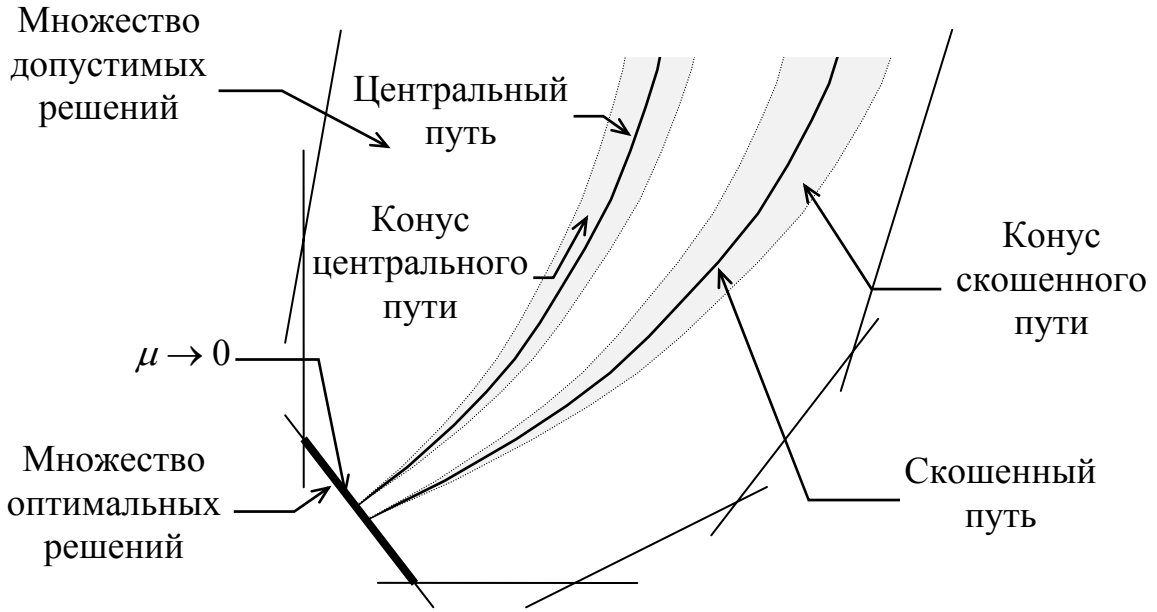

Рис.1. Траектория алгоритмов скошенного пути

Поскольку любая пара векторов $\left(\mathbf{x^1}, \mathbf{u^1}\right)$, лежащих в относительной внутренности допустимого множества, принадлежит скошенному пути, инициируемому вектором с компонентами $t_j = x_j^1 g_j\left(\mathbf{u^1}\right)$, то проблема инициализации для алгоритмов скошенного пути существенно ослабляется.

Идейной основой прямых алгоритмов скошенного пути является приближенное решение методом Ньютона задачи

$$\mathbf{c^T x} - \mu \sum_{j=1}^{n} t_j \ln x_j \to \min, \quad \mathbf{Ax} = \mathbf{b}.$$

Двойственные алгоритмы строятся как процедуры решения методом Ньютона задачи

$$\mathbf{b^T u} + \mu \sum_{j=1}^{n} t_j \ln g_j \to \max, \quad \mathbf{g} + \mathbf{A^T u} = \mathbf{c}.$$

Прямо-двойственные алгоритмы объединяют в себе оба подхода. Отметим, что данные задачи связаны по условиям оптимальности: множители Лагранжа ограничений одной из них являются оптимальными значениями переменных другой.

Предлагается и обосновывается несколько вариантов алгоритмов, являющихся развитием и обобщением разработанных В.И.Зоркальцевым [8], [9] алгоритмов оптимизации в конусе центрального пути. В простейшем из них итеративный переход осуществляется по правилам

$$\mathbf{u^{k+1}} = \arg\min_{\mathbf{u} \in \mathbf{R^m}} \Phi_2\left(\mathbf{x^k}, \mathbf{u}, \mu^k\right) = \left(\mathbf{A X_k^2 M_k^{-1} A^T}\right)\left(\mathbf{A X_k^2 M_k^{-1} c} - \mathbf{b}\right), \qquad (1.2)$$

$$x_j^{k+1} = 2x_j^k - \frac{1}{\mu^k t_j}\left(x_j^k\right)^2 g_j\left(\mathbf{u^{k+1}}\right), \ j = 1, \ldots, n, \qquad (1.3)$$

$$\mu^{k+1} = \left(1 - \frac{\sqrt{\theta(1-\theta)n\gamma} - \theta}{n\gamma - \theta}\right)\mu^k. \qquad (1.4)$$

Здесь и далее $\mathbf{X_k} = diag\left\{x_j^k\right\}$, $\mathbf{M_k} = diag\left\{\mu^k t_j\right\}$.

Теорема 2 [13]. *Для алгоритма* (1.2)–(1.4) *из выполнения условий* (1.1) *на первой итерации следует выполнение условий* (1.1) *на последующих итерациях.*

Полиномиальность алгоритма здесь обеспечивается [6] итеративным уменьшением параметра μ в соответствии с (1.4).

Алгоритм (1.2)–(1.4), обозначаемый далее как алгоритм А, действует в точном соответствии с теоретической оценкой, поэтому с ним (как с эталоном) можно сопоставлять последующие варианты алгоритмов с целью подтверждения эффективности используемых в них процедур, ускоряющих сходимость.

Начиная с алгоритма С параметр скошенного пути μ будет гарантированно уменьшаться на каждой итерации $k > 1$ в соответствии с неравенством

$$\mu^{k+1} \le \left(1 - \sqrt{\frac{\theta(1-\theta)}{n\gamma - \theta}}\right)\mu^k . \qquad (1.5)$$

Поскольку при любых натуральных n и $\theta \in (0;1)$

$$1 - \sqrt{\frac{\theta(1-\theta)}{n\gamma - \theta}} < 1 - \sqrt{\frac{\theta(1-\theta)}{n\gamma}} < 1 - \frac{\sqrt{\theta(1-\theta)n\gamma} - \theta}{n\gamma - \theta},$$

последующие алгоритмы сходятся гарантированно быстрее, чем алгоритм А. Выражение, стоящее в центральной части неравенства, можно использовать в качестве оценки гарантированной скорости сходимости алгоритмов. Наилучшее значение оценки достигается, когда максимально значение величины $\theta(1-\theta)$, т. е. при $\theta = 0.5$. В этом случае

$$\sqrt{\theta(1-\theta)}\Big/\sqrt{n\gamma} = 0.5\Big/\sqrt{n\gamma} .$$

Отсюда получаем, что в предлагаемых алгоритмах уменьшение параметра скошенного пути может осуществляться в соответствии с неравенством

$$\mu^{k+1} < \left(1 - 0.5\Big/\sqrt{n\gamma}\right)\mu^k .$$

Это превосходит оценки, выполняющиеся для алгоритмов следования цели [12], где величина γ определяется как отношение максимальной и минимальной компонент инициирующего вектора, а также (в частном случае при $\gamma = 1$) является наилучшей оценкой для класса алгоритмов оптимизации в конусе центрального пути.

Усовершенствование алгоритмов связано с введением параметрических функций. Идея алгоритма С состоит в том, что закрепляется вектор $\mathbf{x^k}$ переменных прямой задачи и вводится линейная вектор-функция

$$\mathbf{u^k}(\lambda) = \arg\min_{\mathbf{u}\in\mathbf{R^m}} \Phi_2\left(\mathbf{x^k}, \mathbf{u}, \lambda\mu^k\right) = \left(\mathbf{AX_k^2M_k^{-1}A^T}\right)^{-1}\left(\mathbf{AX_k^2M_k^{-1}c} - \lambda\mathbf{b}\right). \qquad (1.6)$$

λ^k находится как решение задачи

$$\lambda \to \min_{\lambda \ge 0}, \quad \Phi_2\left(\mathbf{x}^{\mathbf{k}}, \mathbf{u}^{\mathbf{k}}(\lambda), \lambda\mu^k\right) \le \theta\lambda\mu^k t_{\min}. \tag{1.7}$$

Итеративный переход осуществляется по формулам

$$\mathbf{u}^{\mathbf{k+1}} = \mathbf{u}^{\mathbf{k}}\left(\lambda^k\right),\ \mu^{k+1} = \lambda^k\mu^k,\ x_j^{k+1} = 2x_j^k - \frac{1}{\mu^{k+1}t_j}\left(x_j^k\right)^2 g_j\left(\mathbf{u}^{\mathbf{k+1}}\right). \tag{1.8}$$

Теорема 3 [11]. *Для алгоритма* (1.6)–(1.8) *из выполнения условий* (1.1) *на первой итерации следует выполнение условий* (1.1) *на последующих итерациях. При* $k > 1$ *выполняется неравенство* (1.5).

Алгоритм D является двойственным аналогом алгоритма C. Вычислительный процесс в нем осуществляется по правилам

$$\mathbf{x}^{\mathbf{k}}(\lambda) = \arg \min_{\mathbf{x}\in\mathbf{R}^{\mathbf{n}}:\mathbf{Ax}=\mathbf{b}} \Phi_2\left(\mathbf{x}, \mathbf{u}^{\mathbf{k}}, \lambda\mu^k\right) = \mathbf{G}_{\mathbf{k}}^{-\mathbf{2}}\mathbf{T}\mathbf{A}^{\mathbf{T}}\mathbf{r}^{\mathbf{k}}(\lambda) + \lambda\mu^k\mathbf{G}_{\mathbf{k}}^{-\mathbf{1}}\mathbf{t}. \tag{1.9}$$

где

$$\begin{aligned} &\mathbf{r}^{\mathbf{k}}(\lambda) = \left(\mathbf{A}\mathbf{G}_{\mathbf{k}}^{-\mathbf{2}}\mathbf{T}\mathbf{A}^{\mathbf{T}}\right)^{-\mathbf{1}}\left(\mathbf{b} - \lambda\mu^k\mathbf{A}\mathbf{G}_{\mathbf{k}}^{-\mathbf{1}}\mathbf{t}\right), \\ &\mathbf{G}_{\mathbf{k}} = diag\left\{g_j\left(\mathbf{u}^{\mathbf{k}}\right)\right\},\ \mathbf{T} = diag\left\{t_j\right\}. \end{aligned} \tag{1.10}$$

λ^k определяется как решение задачи

$$\lambda \to \min_{\lambda \ge 0}, \quad \Phi_2\left(\mathbf{x}^{\mathbf{k}}(\lambda), \mathbf{u}^{\mathbf{k}}, \lambda\mu^k\right) \le \theta\lambda\mu^k t_{\min}. \tag{1.11}$$

После чего происходит итеративный переход:

$$\mathbf{x}^{\mathbf{k+1}} = \mathbf{x}^{\mathbf{k}}\left(\lambda^k\right),\ \mu^{k+1} = \lambda^k\mu^k,\ \mathbf{r}^{\mathbf{k+1}} = \mathbf{r}^{\mathbf{k}}\left(\lambda^k\right),\ u_i^{k+1} = u_i^k + \frac{1}{\mu^{k+1}}r_i^{k+1}. \tag{1.12}$$

Теорема 4 [11]. *Для алгоритма* (1.9)–(1.12) *из выполнения условий* (1.1) *на первой итерации следует выполнение условий* (1.1) *на последующих итерациях. При* $k > 1$ *выполняется неравенство* (1.5).

Алгоритм E объединяет в себе вычисление переменных двойственной задачи по правилам алгоритма C и вычисление переменных прямой задачи по правилам алгоритма D и может быть назван прямо-двойственным.

Алгоритмы C_p, D_p и E_p отличаются от C, D и E использованием более высоких степеней p = 4, 8, 16,... при решении вспомогательных задач нахождения величины λ^k. В алгоритме C_p вместо (1.7) решается задача

$$\lambda \to \min_{\lambda \ge 0}, \quad \Phi_p\left(\mathbf{x}^{\mathbf{k}}, \mathbf{u}^{\mathbf{k}}(\lambda), \lambda\mu^k\right) \le \theta^{p/2}\lambda^{p/2}\left(\mu^k\right)^{p/2}\left(t_{\min}\right)^{p/2},$$

где

$$\Phi_p(\mathbf{x},\mathbf{u},\mu)=\sum_{j=1}^{n}\frac{1}{\mu^{p/2}\left(t_j\right)^{p/2}}\left(\mu t_j - x_j g_j(\mathbf{u})\right)^p,$$

а в алгоритме D_p задача (1.11) заменяется на следующую:

$$\lambda\to\min_{\lambda\ge 0},\quad \Phi_p\left(\mathbf{x}^{\mathbf{k}}(\lambda),\mathbf{u}^{\mathbf{k}},\lambda\mu^k\right)\le\theta^{p/2}\lambda^{p/2}\left(\mu^k\right)^{p/2}\left(t_{\min}\right)^{p/2}.$$

Особый интерес представляет использование $p\to\infty$.

Имеется [14] теоретическое обоснование алгоритмов C_p, D_p и E_p для $p=4$. В то же время алгоритмы C_p, D_p и E_p при $p>4$, как показало экспериментальное исследование, успешно решали практические задачи, демонстрируя при этом наивысшую среди всех алгоритмов скорость сходимости.

2. ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ

Важным является практическое использование предложенных выше алгоритмов. Поэтому интересно сравнить их по скорости с вариантами аффинно-масштабирующего метода. Рассматриваемые два варианта F_1 и F_2 построены на основе весовых коэффициентов, формируемых соответственно по правилам (0.3) и (0.4).

Все алгоритмы, кроме прямо-двойственных, имеют приблизительно одинаковый объем вычислений на каждой итерации – наибольшую сложность представляет обращение симметричной положительно определенной матрицы размерности $m\times m$. Поэтому данные о числе итераций могут быть использованы в качестве сравнительной характеристики объема вычислений. На каждой итерации прямо-двойственных алгоритмов используются два обращения матрицы. Следовательно, итерацию этих алгоритмов можно приближенно считать в два раза более трудоемкой.

Итеративный процесс для всех алгоритмов начинался с приближения, полученного в соответствии с процедурой инициализации, изложенной Р. Монтейро и И. Адлером в [6], а завершался с достижением невязкой двойственности $\sum x_j g_j(\mathbf{u})$ заданного критического значения $\varepsilon=0.000005$. Поскольку процедура инициализации предоставляет точку центрального пути, а также в целях нивелирования влияния побочных факторов, сопоставление проводилось на алгоритмах центрального пути – частном случае

предложенных. В то же время выявленные тенденции можно распространить и на весь класс алгоритмов скошенного пути.

Основные результаты вычислительного эксперимента сведены в табл. 1. Величина θ для алгоритма А принималась равной 0.5. Для остальных алгоритмов через знак дроби запишем данные, соответствующие $\theta = 0.5$ и $\theta = 0.9$. Для задачи 4 размерности 100×200 значение θ равнялось 0.9 для всех алгоритмов, кроме А.

Таблица 1. Число итераций, необходимое для решения задач линейного программирования различными алгоритмами

Алг.	**Задача 1** 2×4	**Задача 2** 3×6	**Данцига** 6×12	**Задача 3** 8×18	**Данцига** 19×38	**Задача 4** 19×38	**Задача 4** 100×200
A	90	218	278	258	730	805	2174
C	28/20	50/37	85/63	82/59	260/189	240/170	501
E	16/12	42/33	57/44	55/43	157/120	123/95	193
$\mathbf{C_4}$	25/17	45/29	62/45	50/42	119/101	110/79	142
$\mathbf{E_4}$	14/11	34/27	39/30	34/26	82/64	79/61	95
$\mathbf{C_{16}}$	24/12	44/29	53/39	44/33	99/75	84/53	74
$\mathbf{C_\infty}$	23/12	43/26	52/39	41/31	93/73	78/49	63
$\mathbf{F_1}$	16	35	36	30	62	66	84
$\mathbf{F_2}$	16	37	40	33	69	68	100

Второй эксперимент посвящен непосредственной проверке алгоритмов скошенного пути. Сопоставление проводилось на двух вариантах алгоритма C_4, наиболее эффективного из обоснованных. В первом из них в соответствии с процедурой инициализации из [6] формировалась расширенная задача и осуществлялся процесс оптимизации в конусе центрального пути, а во втором – находилась любая допустимая точка и начинался процесс оптимизации в конусе скошенного пути.

Поскольку меньшие значения коэффициента скошенности способствуют ускорению сходимости вычислительного процесса, предлагается осуществлять по итерациям переход из конуса скошенного пути K1 в конус другого – K2, обладающего меньшим коэффициентом скошенности.

Схематично такой переход изображен на рис. 2. Здесь полученное на *k*-итерации приближение ($\mathbf{x}^{\mathbf{k}}$, $\mathbf{u}^{\mathbf{k}}$) принадлежит обоим конусам – К1 и К2, при этом коэффициент скошенности пути, для которого построен конус К1, меньше аналогичного для конуса К2.

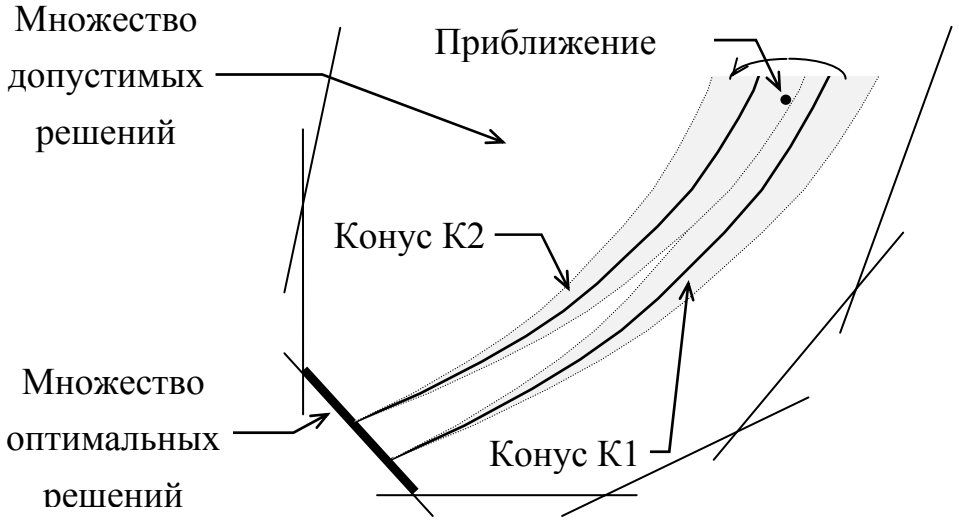

Рис.2. Процедура сокращения коэффициента скошенности

Наиболее простая процедура сокращения коэффициента скошенности состоит в уменьшении тех компонент инициирующего вектора, значения которых превышают $x_j^k g_j\left(\mathbf{u}^{\mathbf{k}}\right)\big/\mu^k$, и одновременном увеличении его минимальной компоненты на максимально возможную величину при условии неувеличения среднего значения и невыхода из конуса скошенного пути. То есть на каждой итерации после получения новых приближений $\mathbf{x}^{\mathbf{k}}$ и $\mathbf{u}^{\mathbf{k}}$, а также нового значения μ^k будем искать новый инициирующий вектор $\widetilde{\mathbf{t}} = \widetilde{\mathbf{t}}(\Delta)$ путем решения следующей задачи:

$$\Delta \to \max,$$

$$\sum_{j=1}^{n} \frac{1}{\mu^k \widetilde{t}_j(\Delta)} \left(\mu^k \widetilde{t}_j(\Delta) - x_j^k g_j\left(\mathbf{u}^{\mathbf{k}}\right)\right)^2 \le \theta \mu^k \left(t_{\min} + \Delta\right),$$

$$\sum_{j=1}^{n} \widetilde{t}_j(\Delta) \le \sum_{j=1}^{n} t_j(\Delta),$$

$$\widetilde{t}_j(\Delta) = \max\left\{t_{\min} + \Delta, \min\left\{t_j, \left(x_j^k g_j\left(\mathbf{u}^{\mathbf{k}}\right)\right)\big/\mu^k\right\}\right\}.$$

Для тестирования были сгенерированы наборы случайных задач различной размерности (по 5 каждой). В табл. 2 внесено среднее количество

итераций, потребовавшееся методам для их решения. Также для сведения приведено среднее значение коэффициента скошенности в стартовой точке, а также среднее его значение на 40-й итерации.

Таблица 2. Среднее число итераций, необходимое для решения случайно сгенерированных задач линейного программирования

Размерность задач	$\mathbf{20 \times 40}$	$\mathbf{50 \times 100}$	$\mathbf{100 \times 200}$	$\mathbf{300 \times 1000}$
Алгоритмы центрального пути	63.6	112.8	–	–
Алгоритмы скошенного пути	64.6	84.0	98.8	194.0
Начальное значение. γ	688.591	4063.856	7256.781	14601.303
Значение γ на 40-й итерации	1.040	2.008	2.970	21.910

Видим, что алгоритмы скошенного пути успешно решили все задачи, в том числе с тысячью переменными. В алгоритмах же центрального пути начали сказываться негативные эффекты, вызываемые процедуры инициализации. Однако существуют задачи, где начальное приближение может быть априори известным, более того, для них может иметься стартовая точка, принадлежащая центральному пути.

3. ИССЛЕДОВАНИЕ СИСТЕМ ЛИНЕЙНЫХ НЕРАВЕНСТВ

Перейдем к рассмотрению систем линейных уравнений и неравенств с интервальными ограничениями на переменные $\mathbf{x} \in \mathbf{R}^{\mathbf{n}}$, $\mathbf{y} \in \mathbf{R}^{\mathbf{m}}$

$$\mathbf{Ax} - \mathbf{y} = \mathbf{0}, \quad \overline{\mathbf{x}} \geq \mathbf{x} \geq \underline{\mathbf{x}}, \quad \overline{\mathbf{y}} \geq \mathbf{y} \geq \underline{\mathbf{y}}. \tag{3.1}$$

Здесь $\mathbf{A} \in \mathbf{R}^{\mathbf{m} \times \mathbf{n}}$ – заданная матрица, $\overline{\mathbf{x}}, \underline{\mathbf{x}} \in \mathbf{R}^{\mathbf{n}}$ и $\overline{\mathbf{y}}, \underline{\mathbf{y}} \in \mathbf{R}^{\mathbf{m}}$ – заданные векторы ограничений, причем $\overline{\mathbf{x}} > \underline{\mathbf{x}}$, $\overline{\mathbf{y}} > \underline{\mathbf{y}}$. Необходимо отыскать допустимое решение системы (3.1) либо максимально быстро идентифицировать ее несовместность.

Система (3.1) с помощью введения дополнительной переменной сводится к задаче линейного программирования. Пусть имеется стартовая точка $\left(\mathbf{x}^{\mathbf{1}}, \mathbf{y}^{\mathbf{1}}\right)$, удовлетворяющая в строгой форме ограничениям-неравенствам, такая, что $\mathbf{Ax}^{\mathbf{1}} \neq \mathbf{y}^{\mathbf{1}}$. Вводится вектор невязок ограничений-равенств

$$\mathbf{r} = \mathbf{y}^1 - \mathbf{A}\mathbf{x}^1$$

и рассматривается задача линейного программирования

$$\alpha \to \min, \quad \mathbf{A}\mathbf{x} - \mathbf{y} + \alpha\,\mathbf{r} = \mathbf{0}, \quad \overline{\mathbf{x}} \ge \mathbf{x} \ge \underline{\mathbf{x}}, \quad \overline{\mathbf{y}} \ge \mathbf{y} \ge \underline{\mathbf{y}}, \quad \alpha \ge 0. \qquad (3.2)$$

Несложно убедиться, что если для оптимального решения задачи (3.2) $\alpha = 0$, то соответствующая пара векторов (**x**,**y**) является допустимой для системы (3.1). Если же для оптимального решения задачи (3.2) значение α положительно, то система (3.1) несовместна.

Одной из наиболее существенных проблем долгое время являлась невозможность быстрой идентификации случая несовместности ограничений системы (3.1). В ранее использовавшихся алгоритмах для этого требовался большой объем вычислений, сопоставимый с необходимым для получения допустимой точки в случае совместности ограничений. В то же время необходимость решения систем линейных уравнений и неравенств часто возникает при итеративной линеаризации существенно нелинейных моделей (в частности, разработанных в ИСЭМ СО РАН моделей управления режимами и идентификации состояния электроэнергетических систем). Из этого следует, что система (3.1) представляет собой линеаризованную подзадачу, нередко оказывающуюся несовместной по причине погрешностей линеаризации, и целесообразно не решать ее до конца, что может оказаться трудоемким процессом, а максимально быстро перейти в новую точку, где заново провести линеаризацию.

В.И.Зоркальцевым на основе теоремы Фаркаша об альтернативных неравенствах был построен и обоснован [15] следующий критерий несовместности:

Теорема 5. *Система уравнений и неравенств* (3.1) *несовместна в том и только в том случае, если существует вектор* $\mathbf{u} \in \mathbf{R}^{\mathbf{m}}$*, при котором*

$$\psi(\mathbf{u}) \equiv \overline{\mathbf{y}}^{\mathbf{T}}\mathbf{u}_- + \underline{\mathbf{y}}^{\mathbf{T}}\mathbf{u}_+ - \overline{\mathbf{x}}^{\mathbf{T}}\left(\mathbf{A}^{\mathbf{T}}\mathbf{u}\right)_+ - \underline{\mathbf{x}}^{\mathbf{T}}\left(\mathbf{A}^{\mathbf{T}}\mathbf{u}\right)_- > 0.$$

Здесь и далее $\left(\mathbf{u}_+\right)_j = \max\left\{0, u_j\right\}$, $\left(\mathbf{u}\_\right)_j = \min\left\{0, u_j\right\}$.

Одной из целей работы являлось включение в алгоритмы и практическая проверка процедуры быстрой идентификации, на основе приведенного критерия, случая несовместности. Другой целью работы была провер-

ка эффективности работы полиномиальных алгоритмов на системах неравенств с совместными ограничениями.

Центрируя и нормируя переменные по формулам

$$x'_j = 2\frac{x_j - \underline{x}_j}{\bar{x}_j - \underline{x}_j},\ j = 1,\ldots,n, \qquad y'_i = 2\frac{y_i - \underline{y}_i}{\bar{y}_i - \underline{y}_i},\ i = 1,\ldots,m,$$

сведем систему (3.1) к следующей:

$$\mathbf{A}'\mathbf{x}' - \mathbf{y}' = \mathbf{b}, \quad 2\mathbf{e} \ge \mathbf{x}' \ge \mathbf{0}, \quad 2\mathbf{e} \ge \mathbf{y}' \ge \mathbf{0}. \tag{3.3}$$

Здесь $\mathbf{e}$ – вектор из единиц соответствующей размерности, элементы матрицы $\mathbf{A}'$ и вектора $\mathbf{b}$ вычисляются по формулам

$$a'_{ij} = a_{ij}\frac{\bar{x}_j - \underline{x}_j}{\bar{y}_i - \underline{y}_i},\ i = 1,\ldots,m,\ j = 1,\ldots,n, \qquad b_i = 2\frac{\underline{y}_i - \sum_{j=1}^{n} a_{ij}\underline{x}_j}{\bar{y}_i - \underline{y}_i},\ i = 1,\ldots,m.$$

Для упрощения записи опустим штрихи над переменными. Введем векторы вспомогательных переменных $\mathbf{s}$ и $\mathbf{q}$ и перейдем к паре взаимнодвойственных задач линейного программирования

$$\begin{array}{l|l} \alpha \to \min, & \\ \mathbf{Ax} - \mathbf{y} + \alpha\mathbf{r} = \mathbf{b}, & \mathbf{u} \\ -\mathbf{x} - \mathbf{s} = -\mathbf{2e}, & \mathbf{v} \\ -\mathbf{y} - \mathbf{q} = -\mathbf{2e}, & \mathbf{w} \\ \mathbf{x} \ge \mathbf{0},\ \mathbf{s} \ge \mathbf{0},\ \mathbf{y} \ge \mathbf{0},\ \mathbf{q} \ge \mathbf{0},\ \alpha \ge 0. & \end{array} \qquad \begin{array}{l|l} \mathbf{b}^{\mathbf{T}}\mathbf{u} - 2\mathbf{e}^{\mathbf{T}}\mathbf{v} - 2\mathbf{e}^{\mathbf{T}}\mathbf{w} \to \max, & \\ \mathbf{g_1} = \mathbf{v} - \mathbf{A}^{\mathbf{T}}\mathbf{u} \ge \mathbf{0}, & \mathbf{x} \\ \mathbf{g_2} = \mathbf{v} \ge \mathbf{0}, & \mathbf{s} \\ \mathbf{g_3} = \mathbf{u} + \mathbf{w} \ge \mathbf{0}, & \mathbf{y} \\ \mathbf{g_4} = \mathbf{w} \ge \mathbf{0}, & \mathbf{q} \\ g_5 = 1 - \mathbf{r}^{\mathbf{T}}\mathbf{u} \ge 0. & \alpha \end{array}$$

Здесь $\mathbf{r} = \mathbf{b} - \mathbf{Ax^1} + \mathbf{y^1}$ при некоторых $\mathbf{x^1} \in (\mathbf{0};\mathbf{2e})$, $\mathbf{y^1} \in (\mathbf{0};\mathbf{2e})$. Справа от черты выписаны двойственные оценки ограничений.

Благодаря особой структуре задачи, имеется начальное приближение, являющееся точкой центрального пути:

$$x_j^1 = s_j^1 = 1,\ j = 1,\ldots,n, \qquad y_i^1 = q_i^1 = 1,\ i = 1,\ldots,m, \qquad \alpha^1 = 1,$$
$$u_i^1 = 0,\ i = 1,\ldots,m, \qquad v_j^1 = 1,\ j = 1,\ldots,n, \qquad w_i^1 = 1,\ i = 1,\ldots,m.$$

Очевидно, что вместе с $\mu^1 = 1$ это приближение удовлетворяет условию

$$x_j\left(\mathbf{v} - \mathbf{A}^{\mathbf{T}}\mathbf{u}\right)_j = \mu,\ s_j v_j = \mu,\ y_i\left(\mathbf{u} + \mathbf{w}\right)_i = \mu,\ q_i w_i = \mu,\ \alpha\left(1 - \mathbf{r}^{\mathbf{T}}\mathbf{u}\right) = \mu.$$

В вычислительную схему алгоритмов включается следующий пункт: если на некоторой итерации выполняется условие

$$\phi\left(\mathbf{u^k}\right) \equiv \mathbf{b}^{\mathbf{T}}\mathbf{u^k} + 2\mathbf{e}^{\mathbf{T}}\mathbf{u}_-^{\mathbf{k}} - 2\mathbf{e}^{\mathbf{T}}\left(\mathbf{A}^{\mathbf{T}}\mathbf{u^k}\right)_+ > 0,$$

то система (3.3) несовместна.

Завершение вычислительного процесса в случае совместности ограничений также формализуется. На каждой итерации проверяется, нельзя ли продвинуться вдоль направления корректировки переменных прямой задачи с шагом $\gamma^k = -\alpha^k / \Delta\alpha^k$, т. е. нарушается ли хоть одно ограничение-неравенство в точке

$$\mathbf{x^{k+1}} = \mathbf{x^k} + \gamma^k \Delta\mathbf{x^k}, \quad \mathbf{s^{k+1}} = \mathbf{s^k} + \gamma^k \Delta\mathbf{s^k},$$

$$\mathbf{y^{k+1}} = \mathbf{y^k} + \gamma^k \Delta\mathbf{y^k}, \quad \mathbf{q^{k+1}} = \mathbf{q^k} + \gamma^k \Delta\mathbf{q^k}.$$

Если все они выполняются, то имеем допустимую точку системы (3.3).

Эксперименты проводились на 14 совместных и 14 несовместных задачах, полученных из моделей управления режимами электроэнергетических систем. Критерий идентифицировал несовместность каждой из 14 несовместных задач на первой же итерации. Результаты расчетов по 14 совместным задачам следующие:

Таблица 3. Число итераций, необходимое для получения решения алгоритмами центрального пути

Алгоритм	**C**	**C_4**	**C_{16}**	**E_4**	**E_{16}**
Число итераций	16.9 (5–26)	10.1 (3–14)	8 (2–11)	7.5 (3–10)	4.8 (2–6)

Представлено среднее количество итераций, потребовавшееся каждому алгоритму для решения задач. В скобках указано минимальное и максимальное число итераций, необходимых для решения отдельной задачи.

Несмотря на хорошие результаты, показанные алгоритмами центрального пути на системах линейных неравенств, алгоритмы скошенного пути оказываются очень актуальными в случае, когда имеется стартовая точка, предполагающаяся в качестве хорошего приближения к решению системы. В частности, при итеративной линеаризации решение, полученное на предыдущей итерации, может с учетом небольшой корректировки рассматриваться в качестве стартового приближения на последующей. Поскольку алгоритмы скошенного пути могут стартовать с любых относительно внутренних точек множеств допустимых решений, представляется

возможным начинать вычислительный процесс именно с него, а не с автоматически формируемого приближения.

ЗАКЛЮЧЕНИЕ

Подытожим результаты проведенного исследования.

1. Полиномиальные алгоритмы решения задач линейного программирования и систем линейных неравенств доведены по эффективности до лучших из используемых алгоритмов.

2. Это произошло за счет отрыва от гарантированной оценки, что показывает значительное (до десятков раз) превосходство наиболее эффективных алгоритмов над исходным вариантом, действующим строго в соответствии с оценкой.

3. Увеличение эффективности работы алгоритмов связано с использованием параметризаций и расширением конуса пути, реализуемым двумя различными способами: увеличением радиуса конуса и использованием более высоких степеней p при решении вспомогательной задачи.

4. Дополнительным преимуществом разработанных алгоритмов скошенного пути является возможность начинать вычислительный процесс с любых относительно внутренних точек множеств допустимых решений, что значительно ослабляет проблему инициализации. Также построена эффективная процедура уменьшения коэффициента скошенности, которая существенно ускоряет вычислительный процесс.

5. Разработаны специальные модификации алгоритмов для решения систем неравенств с двухсторонними ограничениями на переменные, в которых учтены их полезные свойства, такие как возможность быстрой идентификации случая несовместности и возможность получения хорошей стартовой точки для алгоритмов центрального пути.

6. Применение алгоритмов скошенного пути для решения систем неравенств оправдано в случае, если наличествует стартовая точка, подозреваемая в качестве хорошего приближения к решению системы. Это, в частности, происходит при использовании методов последовательной линеаризации.

СПИСОК ЛИТЕРАТУРЫ

1. **Frisch K.** The logarithmic potential method for solving linear programming problems // Memorandum, University Institute of Economics. – Oslo, 1955.
2. **Фиакко А., Мак-Кормик Г.** Нелинейное программирование. Методы последовательной безусловной оптимизации. – М.: Мир, 1972. – 240 с.
3. **Karmarkar N.** A new polynomial-time algorithm for linear programming" // Combinatorica. – 1984. – ¹ 4. – P. 373–395.
4. **Дикин И.И.** Итеративное решение задач линейного и квадратичного программирования // Докл. АН СССР. – 1967. – Т. 174. – С. 747-748.
5. **Зоркальцев В.И.** Метод относительно внутренних точек. – Сыктывкар: Коми филиал АН СССР. – 1986. – 18 с.
6. **Monteiro R., Adler I.** Interior path-following primal-dual algorithms. Part 1: Linear programming // Mathematical programming. – 1989. – № 44. – P. 27–49.
7. **Kojima M., Mizuno S., Yoshise A.** A polynomial-time algorithm for a class of linear complementarity problems // Mathematical programming. – 1989. – ¹ 44. – P. 1–26.
8. **Зоркальцев В.И., Нечаева М.С.** Оптимизация в конусе центрального пути. Иркутск, 1995. – 24 с. – (Препр. СЭИ СО РАН; № 2).
9. **Зоркальцев В.И.** Алгоритмы оптимизации в конусе центрального пути // Журн. вычисл. математики и мат. физики. – 2000. – Т. 40, № 2. – С. 318–327.
10. **Ye Y., Todd M., Mizuno S.** An O($\sqrt{n}$)-iteration homogeneous and self-dual linear programming algorithm // Mathematics of operation research. – 1994. – ¹ 19. – P. 53–67.
11. **Войтов О.Н., Зоркальцев В.И., Филатов А.Ю.** Алгоритмы скошенного пути для решения задач линейного программирования // Дискретный анализ и исследование операций. – 2001. – Сер. 2, Т. 8, № 2. – С. 17–26.

12. **Jansen B., Roos C., Terlaky T., Vial J.-Ph.** Primal-dual target-following algorithms for linear programming // Annals of operations research. – 1996. – ¹ 62. – P. 197–231.
13. **Филатов А.Ю.** Развитие алгоритмов внутренних точек и их приложение к системам неравенств: Дис. ... канд. физ.-мат. наук. – Иркутск, 2001. – 123 с.
14. **Зоркальцев В.И., Филатов А.Ю.** Новый алгоритм скошенного пути для решения задачи линейного программирования // Тр. XII Байкальской международной конференции “Методы оптимизации и их приложения”. – Иркутск, 2001. – Т. 1. – С. 16–22.
15. **Войтов О.Н., Зоркальцев В.И., Филатов А.Ю.** Исследование систем неравенств алгоритмами внутренних точек на задачах поиска допустимых режимов электроэнергетических систем. – Иркутск, 1997. – 30 с. – (Препр. СЭИ СО РАН; № 10).