БЫСТРЫЙ АЛГОРИТМ РЕШЕНИЯ КВАДРАТИЧНОЙ ЗАДАЧИ О РАНЦЕ*

А. В. Плоткин

avplotkin@gmail.com

24 февраля 2020 г.

Аннотация. В работе рассматривается задача квадратичного программирования со строго выпуклой сепарабельной целевой функцией, единственным линейным ограничением и двусторонними ограничениями на переменные. В англоязычной литературе эта задача называется Convex Knapsack Separable Quadratic Problem, или сокращенно — CKSQP (см., например, диссертацию [1] и библиографию в ней). Нас интересует алгоритм решения задачи CKSQP с линейной сложностью. Посвященные этой теме работы содержат неточности в описании алгоритмов и неэффективные реализации [2]. В данной работе удалось преодолеть имеющиеся трудности.

1°. Рассмотрим следующую задачу квадратичного программирования:

$$\begin{gathered} \tfrac{1}{2}\langle Hx, x\rangle - \langle c, x\rangle \to \min\limits_{x\in\Omega}, \\ \Omega : \begin{cases} \sum\limits_{i=1}^{n} a_i x_i = b, \\ \ell_i \leqslant x_i \leqslant r_i, \quad i \in 1:n. \end{cases} \end{gathered} \tag{1}$$

Здесь b, c_i, a_i — вещественные параметры, $\ell_i \in \mathbb{R}\cup\{-\infty\}$, $r_i \in \mathbb{R}\cup\{+\infty\}$, и H — диагональная матрица порядка n с положительными диагональными элементами $h_1, h_2, \ldots, h_n$. Целевая функция задачи (1) является строго выпуклой и сепарабельной.

В тексте статьи мы будем предполагать, что ℓ_i, $r_i \in \mathbb{R}$. Это позволит упростить изложение основных идей алгоритма. При этом разработанное программное обеспечение будет учитывать случаи $\ell_i = -\infty$ и $r_i = +\infty$.

*Семинар по конструктивному негладкому анализу и недифференцируемой оптимизации «CNSA & NDO»: `http://www.apmath.spbu.ru/cnsa/`

Во многих приложениях задача (1) встречается с единичной матрицей H, что соответствует проектированию точки c на множество Ω [3, 4]. Из других приложений можно выделить квадратичную задачу распределений ресурсов [5] и вычисление матрицы обновления в квазиньютоновском методе при наличии ограничений на якобиан [6].

2°. Не умаляя общности, можно считать, что в задаче (1) выполнены условия $\ell_i < r_i$ и $a_i \neq 0$ при всех i от 1 до n. Если $a_i = 0$, то оптимальное значение x_i находится путем решения тривиальной задачи:

$$\begin{aligned} &\tfrac{1}{2}h_i x_i^2 - c_i x_i \to \min, \\ &\ell_i \leqslant x_i \leqslant r_i. \end{aligned}$$

Воспользовавшись предположением о том, что коэффициенты a_i в определении множества Ω отличны от нуля, выполним замену переменных:

$$x_i = \hat{x}_i/a_i + c_i/h_i, \quad i \in 1:n. \tag{2}$$

Получим задачу следующего вида:

$$\begin{gathered} \tfrac{1}{2}\langle \hat{H}\hat{x}, \hat{x}\rangle \to \min_{\hat{x}\in\hat{\Omega}}, \\ \hat{\Omega}: \begin{cases} \displaystyle\sum_{i=1}^{n} \hat{x}_i = \hat{b}, \\ \hat{\ell}_i \leqslant \hat{x}_i \leqslant \hat{r}_i, \quad i \in 1:n, \end{cases} \end{gathered} \tag{3}$$

где

$$\begin{aligned} \hat{H} &= \operatorname{diag}(h_1/a_1^2, \ldots, h_n/a_n^2), \\ \hat{b} &= b - \sum_{i=1}^{n} \frac{a_i c_i}{h_i}, \\ \hat{\ell}_i &= \begin{cases} a_i(\ell_i - c_i/h_i), & \text{если} \quad a_i > 0, \\ a_i(r_i - c_i/h_i), & \text{если} \quad a_i < 0, \end{cases} \\ \hat{r}_i &= \begin{cases} a_i(r_i - c_i/h_i), & \text{если} \quad a_i > 0, \\ a_i(\ell_i - c_i/h_i), & \text{если} \quad a_i < 0. \end{cases} \end{aligned}$$

Задача (3) является частным случаем задачи (1). Преобразование (2) имеет линейную сложность и значительно упрощает дальнейшие вычисления. В связи с этим, далее будем считать, что решаемая задача имеет следующий вид:

$$\begin{gathered} Q(x) := \tfrac{1}{2}\langle Hx, x\rangle \to \min_{x\in\Omega}, \\ \Omega: \begin{cases} \displaystyle\sum_{i=1}^{n} x_i = b, \\ \ell_i \leqslant x_i \leqslant r_i, \quad i \in 1:n. \end{cases} \end{gathered} \tag{4}$$

3°. Целевая функция задачи (4) непрерывна и строго выпукла, а значит критерием существования решения является непустота множества планов Ω. В данном случае условие непустоты множества планов записывается следующим образом:

$$\sum_{i=1}^{n} \ell_i \leqslant b \leqslant \sum_{i=1}^{n} r_i. \tag{5}$$

Будем считать, что условие (5) выполнено, а значит множество планов Ω непусто.

Заметим, что множество Ω является выпуклым. Вместе со строгой выпуклостью целевой функции $Q(x)$ это гарантирует единственность решения задачи (4). Для его нахождения воспользуемся условиями Куна-Таккера:

$$\left.\begin{gathered} h_i x_i = \lambda + u_i - v_i, \\ u_i(x_i - \ell_i) = 0, \quad v_i(r_i - x_i) = 0, \\ u_i \geqslant 0, \quad v_i \geqslant 0, \\ \ell_i \leqslant x_i \leqslant r_i, \end{gathered}\right\} \quad i \in 1:n, \tag{6}$$

$$\sum_{i=1}^{n} x_i = b. \tag{7}$$

4°. Разберемся с условиями (6). Для этого нам потребуется вспомогательная лемма.

ЛЕММА. *Пусть выполнены соотношения*

$$hw = \lambda + u - v, \tag{8}$$

$$u(w - \ell) = 0, \quad v(r - w) = 0, \tag{9}$$

$$u \geqslant 0, \quad v \geqslant 0, \tag{10}$$

$$\ell \leqslant w \leqslant r,$$

где h, ℓ, r — вещественные константы, причем $c > 0$ и $\ell < r$. Тогда необходимо

$$w = \begin{cases} \ell, & \text{если } \lambda/h \leqslant \ell, \\ \lambda/h, & \text{если } \ell < \lambda/h < r, \\ r, & \text{если } \lambda/h \geqslant r. \end{cases} \tag{11}$$

Доказательство основано на анализе полного набора альтернатив для переменной w.

(a1) $w = \ell$. Согласно (9), $v = 0$. В силу (8) и (10) имеем $hw = \lambda + u \geqslant \lambda$, так что

$$\lambda \leqslant h\ell. \tag{12}$$

(a2) $\ell < w < r$. Согласно (9), $u = v = 0$, так что в силу (8) имеем $\lambda = hw$. В частности,

$$\lambda \in (h\ell, hr). \tag{13}$$

(a3) $w = r$. Согласно (9), $u = 0$. В силу (8) и (10) имеем $hw = \lambda - v \leqslant \lambda$, так что

$$\lambda \geqslant hr. \tag{14}$$

Отметим, что условия (12)–(14) представляют собой полный набор альтернатив для переменной $\lambda \in \mathbb{R}$.

Переходим к доказательству формулы (11).

Если $\lambda/h \leqslant \ell$, то $\lambda \notin (h\ell, hr)$ и $\lambda \leqslant h\ell < hr$, поэтому альтернативы (a2) и (a3) исключаются. Остается одна возможность $w = \ell$.

Если $\ell < \lambda/h < r$, то $\lambda > h\ell$ и $\lambda < hr$, поэтому альтернативы (a1) и (a3) исключаются. Остается одна возможность $\ell < w < r$. Согласно (a2), $w = \lambda/h$.

Наконец, если $\lambda/h \geqslant r$, то $\lambda \notin (h\ell, hr)$ и $\lambda \geqslant hr > h\ell$, поэтому альтернативы (a1) и (a2) исключаются. Остается одна возможность $w = r$.

Лемма доказана. □

Формула (11) определяет переменную w как функцию от λ, $w = w(\lambda)$. На рис. 1 изображен график функции $w(\lambda)$.

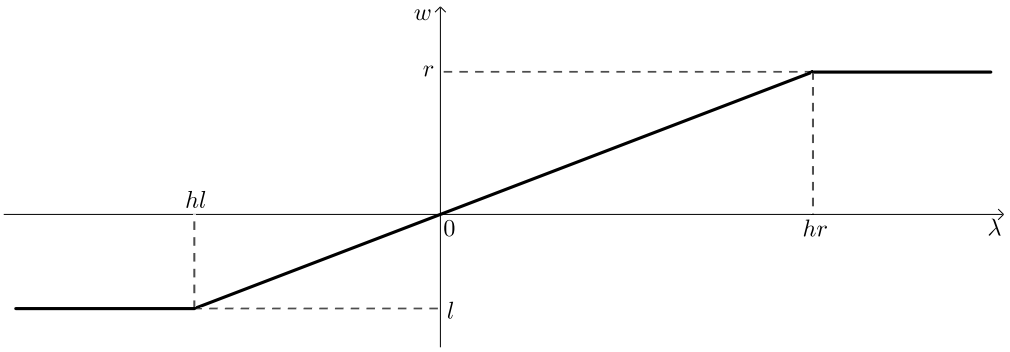

Рис. 1. График функции $w(\lambda)$

Функция $w(\lambda)$ допускает аналитическое представление (см, например, [7])

$$w(\lambda) = \ell + \tfrac{1}{h}(\lambda - h\ell)_+ - \tfrac{1}{h}(\lambda - hr)_+, \tag{15}$$

где $(u)_+ = \max\{0, u\}$.

5°. Вернемся к условиям Куна-Таккера (6), (7).

Воспользуемся доказанной леммой при каждом i от 1 до n. Положив $w = x_i$, $h = h_i$, $\ell = \ell_i$, $r = r_i$, $u = u_i$, $v = v_i$, получим представление x_i как функций от λ:

$$x_i(\lambda) = \ell_i + \tfrac{1}{h_i}(\lambda - h_i\ell_i)_+ - \tfrac{1}{h_i}(\lambda - h_i r_i)_+, \quad i \in 1:n.$$

Каждая из функций $x_i(\lambda)$ является непрерывной, неубывающей, имеет минимальное значение ℓ_i и максимальное r_i.

Таким образом, условию (6) удовлетворяет множество

$$X = \left\{ \big(x_1(\lambda), \ldots, x_n(\lambda)\big) \mid \lambda \in \mathbb{R} \right\}.$$

Добавив необхоимость выполнения условия (7) на этом множестве, получим следующее уравнение:

$$\sum_{i=1}^{n} x_i(\lambda) = b, \quad \lambda \in \mathbb{R}. \tag{16}$$

Свойства функций $x_i(\lambda)$ вместе с условием (5) гарантируют существование решения у данного уравнения.

Подведем итоги.

ТЕОРЕМА. *Пусть λ^* — решение уравнения* (16)*, тогда $x^* = (x_1^*, \ldots, x_n^*)$, где $x_1^* = x_1(\lambda^*), \ldots, x_n^* = x_n(\lambda^*)$, является единственным решением задачи* (4).

Далее нас будет интересовать быстрый алгоритм решения уравнения (16).

6°. Введем следующие обозначения:

$$\left.\begin{aligned} &\alpha_i = \tfrac{1}{h_i}, \\ &\alpha_{i+n} = -\tfrac{1}{h_i}, \\ &\lambda_i = h_i\ell_i, \\ &\lambda_{i+n} = h_i r_i, \end{aligned}\right\} \quad i \in 1:n,$$

$$\varphi_i(\lambda) = \alpha_i(\lambda - \lambda_i)_+, \quad i \in 1:2n,$$

$$F(\lambda) = \sum_{i=1}^{2n} \varphi_i(\lambda),$$

$$C = b - \sum_{i=1}^{n} \ell_i.$$

Тогда уравнение (16) можно переписать в следующем виде:

$$F(\lambda) = C. \tag{17}$$

При этом функция $F(\lambda)$ является неубывающей ломаной с узлами $\lambda_1, \ldots, \lambda_{2n}$.

Во избежание двусмысленности, дадим определение *медиане*, которому будем придерживаться в дальнейшем.

ОПРЕДЕЛЕНИЕ. Пусть имеется конечное множество вещественных чисел $A = \{a_i\}_{i=1}^m$. Пусть $b_1, b_2, \ldots, b_m$ — последовательность, полученная в результате упорядочивания элементов множества A в порядке неубывания. Тогда элемент $b_{\lceil \frac{m+1}{2} \rceil}$ называется *медианой* множества A.

Напомним, что медиану множества A можно найти за линейное время [8, с. 250–253].

Перейдем к построению алгоритма.

Условие существования решения гарантирует нам выполнение следующего неравенства:

$$F(\min_{i \in 1:2n} \lambda_i) \leqslant C \leqslant F(\max_{i \in 1:2n} \lambda_i).$$

Наш алгоритм будет вычислять значения функции F исключительно в узлах λ_i, а в качестве результата будет возвращать узел $\hat{\lambda} = \max\{\lambda_i \mid F(\lambda_i) \leqslant C\}$. Имея такой узел, несложно найти ближайший к нему узел справа (если такой имеется), вычислить значение функции в обоих узлах и выполнить обратную интерполяцию для нахождения λ^*. Такие дополнительные действия требуют линейного времени работы и не окажут влияния на итоговую вычислительную сложность алгоритма.

Для упрощения понимания, сначала мы представим описание алгоритма, работающего за время $O(n \log n)$, а затем покажем, как добиться линейного времени работы.

7°. Опишем упрощенную версию алгоритма.

Начальный шаг. Положим $I_0 = \{1, 2, \ldots, 2n\}$.

Инвариант. Искомый узел $\hat{\lambda}$ содержится в множестве Λ_k, где $\Lambda_k = \{\lambda_i\}_{i \in I_k}$.

Шаг алгоритма. Пусть имеется множество I_k. Если множество I_k содержит единственный элемент, то этот элемент является индексом искомого узла $\hat{\lambda}$. Иначе, найдем медиану множества Λ_k и обозначим этот узел через λ_{med}. Вычислим значение функции: $F_{\text{med}} = F(\lambda_{\text{med}})$. Если $F_{\text{med}} \leqslant C$, то значение $\hat{\lambda}$ точно не меньше λ_{med}. Если же $F_{\text{med}} > C$, то $\hat{\lambda}$ точно меньше λ_{med}.

Таким образом, положим

$$I_{k+1} = \begin{cases} \{\lambda_{\text{med}}\} \cup \{\lambda \in \Lambda_k \mid \lambda > \lambda_{\text{med}}\}, & \text{если } F_{\text{med}} \leqslant C, \\ \{\lambda \in \Lambda_k \mid \lambda < \lambda_{\text{med}}\}, & \text{если } F_{\text{med}} > C. \end{cases}$$

Полученное множество I_{k+1} является подмножеством I_k и имеет размер не превышающий $\left\lceil \frac{|I_k|}{2} \right\rceil$.

Оценка сложности. Алгоритм делает $O(\log n)$ итераций. На каждой итерации алгоритма выполняется вычисление функции F за $O(n)$ операций. Затраты на вычисление медианы суммарно составляют $O(2n)+O(n)+O(\frac{n}{2})+O(\frac{n}{4})+\ldots+O(1)=O(n)$. Таким образом, итоговая сложность алгоритма составляет $O(n\log n)$.

8°. Для достижения линейной сложности необходимо научиться вычислять значение целевой функции как можно быстрее. Добъемся того, чтобы вычисление функции F на k-й итерации требовало $O(|I_k|)$ операций.

Сделаем следующее наблюдение. Возьмем произвольный индекс i_0. Пусть в какой-то момент этот индекс вышел из последовательности множеств I: $i_0 \in I_k$, $i_0 \notin I_{k+1}$. Тогда $\lambda_{i_0} \leqslant \min \Lambda_{k+1}$ или $\lambda_{i_0} \geqslant \max \Lambda_{k+1}$ по построению. Так как все дальнейшие вычисления функции F будут происходить только в узлах I_{k+1}, то уже на этапе исключения индекса i_0 можно однозначно описать вклад функции $\varphi_{i_0}(\lambda)$ в $F(\lambda)$. Если $\lambda_{i_0} \leqslant \min \Lambda_{k+1}$, то $\varphi_{i_0}(\lambda) = \alpha_{i_0}\lambda - \alpha_{i_0}\lambda_{i_0}$ для всех $\lambda \in I_{k+1}$. Если $\lambda_{i_0} \geqslant \max \Lambda_{k+1}$, то $\varphi_{i_0}(\lambda) = 0$ для всех $\lambda \in I_{k+1}$.

Введем и будем поддерживать линейную функцию $A_k\lambda + B_k$, в которой будут «аккумулироваться» линейные функции узлов, вышедших из рассмотрения.

Начальный шаг. Положим $I_0 = \{1, 2, \ldots, 2n\}$, $A_0 = 0$, $B_0 = 0$.

Инвариант 1. Искомый узел $\hat{\lambda}$ содержится в множестве Λ_k, где $\Lambda_k = \{\lambda_i\}_{i\in I_k}$.

Инвариант 2. Для всех узлов λ из I_k значение функции F может быть вычислено по формуле $F(\lambda) = \sum_{i\in I_k} \varphi_i(\lambda) + A_k\lambda + B_k$.

Шаг алгоритма. Пусть имеется множество I_k. Если множество I_k содержит единственный элемент, то этот элемент является индексом искомого узла $\hat{\lambda}$. Иначе, найдем медиану множества Λ_k и обозначим этот узел через λ_{med}. Вычислим значение функции: $F_{\text{med}} = \sum_{i\in I_k} \varphi_i(\lambda_{\text{med}}) + A_k\lambda_{\text{med}} + B_k$. Если $F_{\text{med}} \leqslant C$, то значение $\hat{\lambda}$ не меньше λ_{med}. Если же $F_{\text{med}} > C$, то $\hat{\lambda}$ точно меньше λ_{med}.

В случае $F_{\text{med}} \leqslant C$ положим

$$
\begin{aligned}
I_{k+1} &= \{\lambda_{\text{med}}\} \cup \{\lambda \in \Lambda_k \mid \lambda > \lambda_{\text{med}}\},\\
A_{k+1} &= A_k + \sum_{i\in I_k\setminus I_{k+1}} \alpha_i,\\
B_{k+1} &= B_k - \sum_{i\in I_k\setminus I_{k+1}} \alpha_i\lambda_i.
\end{aligned}
$$

В случае $F_{\text{med}} > C$ положим

$$\begin{aligned} I_{k+1} &= \{\lambda \in \Lambda_k \mid \lambda < \lambda_{\text{med}}\}, \\ A_{k+1} &= A_k, \\ B_{k+1} &= B_k. \end{aligned}$$

Оценка сложности. Теперь затраты на вычисление функций, пересчет коэффициентов и работу со множествами составляют $O(2n)+O(n)+O(\frac{n}{2})+O(\frac{n}{4})+\ldots+O(1)=O(n)$. Таким образом, итоговая сложность алгоритма составляет $O(n)$.

9°. Проведем численные эксперименты.

Будем решать задачи вида (4) для разных n: $10^4, 10^5, 10^6, 10^7$. Коэффициенты h будем выбирать случайно из отрезка $[0.1, 1]$, коэффициенты ℓ и r — из отрезка $[-1, 1]$. Параметр b выберем случайно из отрезка $\left[\sum_{i=1}^n \ell_i, \sum_{i=1}^n r_i\right]$.

В нашем алгоритме для поиска медианы мы будем использовать алгоритм `Qickselect` [8, с. 245–249], осуществляющий поиск за $O(n)$ в среднем, но показывающий хорошие результаты на практике.

Для каждого из рассматриваемых n были сгенерированы 100 тестовых задач. Вычисления производились на машине с процессором AMD Ryzen 5 3550h и 8 гб ОЗУ. В таблице ниже приводится среднее время работы обеих версий алгоритма.

n	Версия $O(n \log n)$	Версия $O(n)$
10^4	2	1
10^5	16	12
10^6	166	113
10^7	1921	1249

Рис. 2. Среднее время работы алгоритмов (в мс)

Реализация алгоритма и результаты численных экспериментов доступны по адресу `https://github.com/arxmath/CKSQP`.

ЛИТЕРАТУРА

1. J. Jeong. *Indefinite Knapsack Separable Quadratic Programming: Methods and Applications*. (2014).

2. K. Kiwiel. *Breakpoint searching algorithms for the continuous quadratic knapsack problem.* Mathematical Programming 112 (2008), с. 473–491.

3. Y. H. Dai, R. Fletcher. *New algorithms for singly linearly constrained quadratic programs subject to lower and upper bounds*. Math. Program. 106 (2006), с. 403–421.

4. S. Nielsen and S. Zenios. *Massively parallel algorithms for singly-constrained convex programs*. ORSA J. Comput., 4 (1992), с. 166–181.

5. M. Patriksson. *A survey on the continuous nonlinear resource allocation problem*. European J. Operational Research. 185 (2008), с. 1–46.

6. P. Calamai and J. Moré. *Quasi-Newton updates with bounds*. SIAM J. Numer. Anal., 24 (1987), с. 1434–1441.

7. В. Н. Малоземов, Г. Ш. Тамасян. *Представления непрерывных кусочно-аффинных функций* // Семинар «CNSA & NDO». Избранные доклады. 10 сентября 2019 г.

8. Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. *Алгоритмы. Построение и анализ* // 3-е издание, Вильямс, 2013.