РЕШЕНИЕ ЗАДАЧИ СИЛЬВЕСТРА В MATLAB*

М. А. Кольцов
kolmax94@gmail.com

26 февраля 2015 г.

1°. Постановка задачи. Будем рассматривать *задачу Сильвестра* — по заданному набору точек $a_i \in \mathbb{R}^n$, $i \in M = 1:m$, найти наименьший по объёму шар, их содержащий. Чтобы формально поставить задачу, введем переменную $x \in \mathbb{R}^n$ — центр искомого шара. Очевидно, что для того, чтобы шар $B_{\mathbb{R}^n}(x, r)$ содержал все заданные точки, необходимо, чтобы его радиус r удовлетворял следующему условию:

$$\forall\, i \in M \ \|a_i - x\| \leqslant r$$

То есть, в задаче Сильвестра требуется найти точку минимума функции $\max\limits_{i\in M}\{\|a_i - x\|\}$ на $\mathbb{R}^n$, при этом радиус искомого минимального шара будет равен значению функции в этой точке. Заметим, что выражение под знаком максимума можно заменить на $\frac{1}{2}\|a_i - x\|^2$, что не меняет точку минимума. Теперь можно сказать так: решение задачи Сильвестра — это точка x_*, являющаяся решением экстремальной задачи

$$\varphi(x) := \max_{i\in M}\{\tfrac{1}{2}\|a_i - x\|^2\} \to \min_{x\in\mathbb{R}^n}, \tag{1}$$

а радиус соответствующего минимального шара равен $\sqrt{2\varphi(x_*)}$. Доказательство существования и единственности такого решения имеется в книге [1].

Задача (1) сводится к задаче квадратичного программирования. Раскроем по определению нормы:

$$\varphi(x) = \tfrac{1}{2}\|x\|^2 + \max_{i\in M}\{-\langle a_i, x\rangle + \tfrac{1}{2}\|a_i\|^2\}$$

Теперь выражение под максимумом линейно по x, а значит сам максимум можно обозначить за t и ввести набор ограничений. То есть, исходная задача

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

эквивалентна задаче квадратичного программирования

$$\frac{1}{2}\langle Ex, x\rangle + t \to \min$$
$$\langle a_i, x\rangle + t \geqslant \frac{1}{2}\|a_i\|^2, \quad i \in M$$

2°. Общий вид задачи квадратичного программирования и двойственной к ней. Чтобы двигаться дальше, необходимо сначала рассмотреть понятия задачи квадратичного программирование и двойственной задачи. Задача квадратичного программирования — это экстремальная задача вида

$$Q(x) = \frac{1}{2}\langle Dx, x\rangle + \langle c, x\rangle \to \min$$
$$A[M_1, N] \times x[N] \geqslant b[M_1]$$
$$A[M_2, N] \times x[N] = b[M_2]$$
$$x[N_1] \geqslant \mathbb{O}[N_1]$$

где $D = D[N, N]$ — симметричная неотрицательно-определённая матрица. Как и в случае линейного программирования, можно ввести двойственную задачу и получить теорему, связывающую её с исходной задачей — первую теорему двойственности. При этом двойственная задача выглядит так:

$$-\frac{1}{2}\langle Dv, v\rangle + \langle b, u\rangle \to \max$$
$$-v[N] \times D[N, N_1] + u[M] \times A[M, N_1] \leqslant c[N_1]$$
$$-v[N] \times D[N, N_2] + u[M] \times A[M, N_2] = c[N_2]$$
$$u[M_1] \geqslant \mathbb{O}[M_1]$$

Для удобства записи двойственной задачи можно использовать таблицу

	c	
v	$-D$	$-\frac{1}{2}Dv$
u	A	b

Тогда целевая функция получается скалярным умножением первого и последнего столбца таблицы, ограничения — первого и второго, а правые части ограничений стоят в первой строке, причём неравенствами являются ограничения, соответствующие знаковым ограничениям исходной задачи. Знаковые ограничения накладываются на двойственные переменные (компоненты вектора u), соответствующие исходным ограничениям-неравенствам.

3°. Двойственная задача для задача Сильвестра и её решение. Вернёмся к задаче Сильвестра. Из векторов-строк a_i составим матрицу A_0 и введём стандартные для квадратичной задачи обозначения: матрицу квадратичной формы D, вектор коэффициентов линейной формы c, матрицу ограничений A и столбец правых частей b. Получаем следующее:

$$D = \begin{pmatrix} & & 0 \\ & E[N,N] & \vdots \\ & & 0 \\ 0 & \cdots \quad 0 & 0 \end{pmatrix} \quad c = \begin{pmatrix} 0 & \cdots & 0 & 1 \end{pmatrix} \quad A = \begin{pmatrix} & & 1 \\ & A_0 & \vdots \\ & & 1 \end{pmatrix} \quad b[i] = \frac{1}{2}\|a_i\|^2$$

Переменной в такой задаче является пара $y = (x, t)$, а в двойственной вектор $z = ((v, s), u)$. Чтобы записать двойственную задачу, воспользуемся таблицей

	$0 \quad \cdots \quad 0 \quad 1$	
$v[N]$	$-E[N,N] \quad \begin{matrix}0\\\vdots\\0\end{matrix}$	$-\frac{1}{2}v[N]$
s	$0 \quad \cdots \quad 0 \quad 0$	0
u	$A_0 \quad \begin{matrix}1\\\vdots\\1\end{matrix}$	b

Тогда по описанному общему алгоритму имеем

$$\begin{gathered} g(z) := -\frac{1}{2}\langle v, v\rangle + \langle b, u\rangle \to \max \\ -v + uA_0 = 0 \\ \sum_{i\in M} u[i] = 1 \\ u[M] \geqslant \mathbb{O}[M] \end{gathered}$$

Видно, что здесь можно исключить вектор v и получить задачу минимизации на стандартном симплексе

$$\begin{gathered} h(u) := \frac{1}{2}\|uA_0\|^2 - \langle b, u\rangle \to \min \\ \sum_{i\in M} u[i] = 1 \\ u[M] \geqslant \mathbb{O}[M] \end{gathered}$$

Для последней задачи существует оптимальный план u_*. В [1] доказано, что в таком случае вектор $x_* = u_*A_0$ является решением задачи Сильвестра. Так

как два оптимальных плана пары двойственных задач имеют одно и то же значение целевой функции, то $\varphi(x_*) = -h(u_*)$ (на последнем шаге целевая функция была умножена на -1 чтобы перейти от задачи максимизации к задаче минимизации). Значит, радиус искомого минимального шара равен $\sqrt{-2h(u_*)}$.

4°. Решение задачи в среде MATLAB. Для удобства решения задачи Сильвестра была написана функция `sylvester`, принимающая единственный аргумент — матрицу A, строки которой образованы исходным множеством точек. Алгоритм, реализованный в функции, следующий:

1) Вычислить матрицу $D = AA^T$ и вектор b, соответствующие двойственной задаче.

2) Вызвать встроенную функцию `quadprog` для решения двойственной задачи и получить её решение — вектор u_*, и значение целевой функции f на нём.

3) Вычислить радиус минимального шара по формуле $r = \sqrt{-2f}$.

Написанная функция работает с пространствами любых размерностей, но для наглядности она была проверена на размерности 2. Для этого генерировались случайные множества точек, лежащих внутри и на границе круга $(x-4)^2 + (y-3)^2 \leqslant 4$ и для них решалась задача Сильвестра. Одно из таких решений с количеством точек 100 представлено на рис. 1.

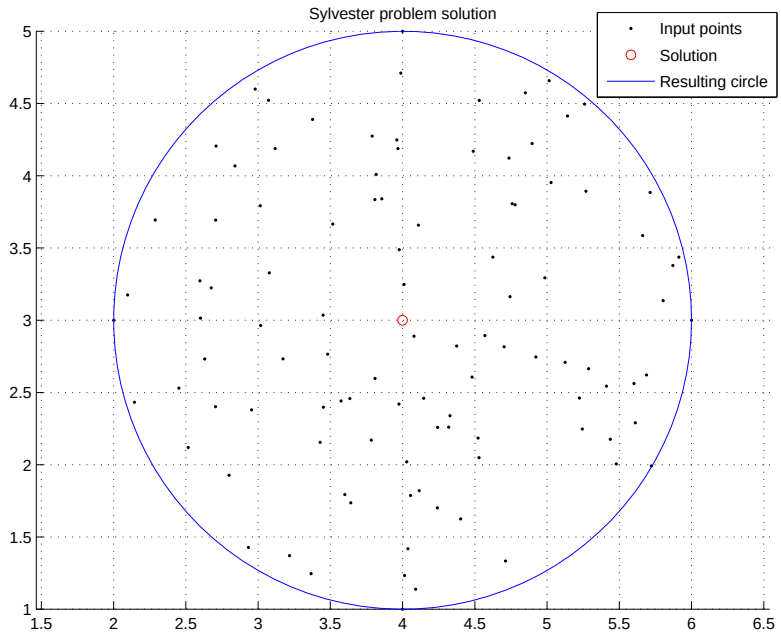

Рис. 1. Решение задачи Сильвестра для $m = 100$

Стоит отметить, что уже при попытке решить задачу для 200 точек, мы получаем ошибку:

```
Maximum number of iterations exceeded; increase options.MaxIter.
To continue solving the problem with the current solution as the
starting point, set x0 = x before calling quadprog.
```

Это связано с тем, что по умолчанию количество итераций, которые делает функция `quadprog`, ограничено числом 200. Чтобы решить эту проблему, можно поднять ограничение. При этом оказывается, что количество итераций зависит от количества точек примерно линейно (см. рис. 2).

Такой результат можно объяснить тем, что по умолчанию MATLAB выбирает для решения этой задачи алгоритм «active-set». Если же задать использование алгоритма «interior-point-convex», пригодного для решения выпуклых задач, то для 100 точек требуется 6 итераций, а для 500 — всего 9.

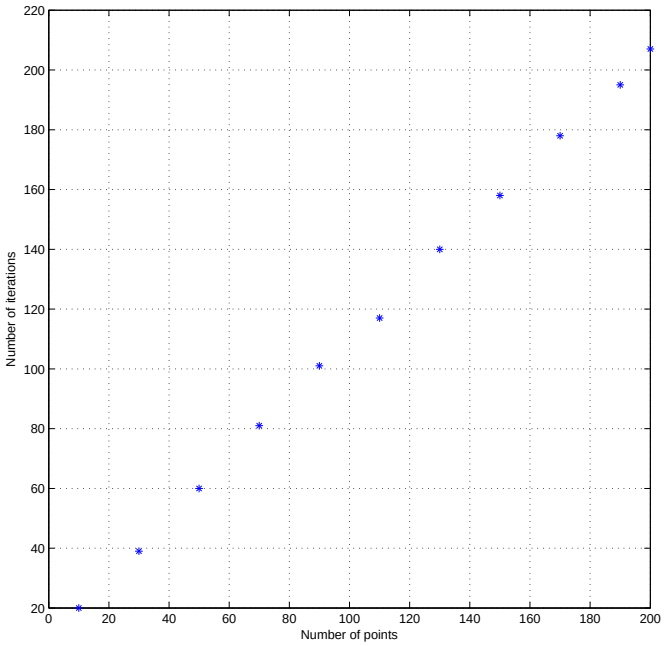

Рис. 2. Зависимость числа итераций от числа точек

Также можно протестировать программу в размерности 3. Для этого было сгенерировано 100 случайных точек внутри и на границе стандартного куба $[-1, 1]^3$. На этом наборе точек программа выдает ожидаемый ответ — центр шара близок к 0, а его радиус — к $\sqrt{3}$.

ЛИТЕРАТУРА

1. Гавурин М. К., Малозёмов В. Н. *Экстремальные задачи с линейными ограничениями.* Л.: Изд-во ЛГУ, 1984. 176 с.