1 Outline

In this lecture, we cover

- linear programming standard form,
- history of linear programming.

2 Linear programming standard form

Remember that our linear program

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b, \\ & x \in \mathbb{R}^d \end{aligned} \tag{5.1}$$

has the constraint system $Ax \leq b$. When A has m rows, i.e. A is an $m \times d$ matrix, Ax is a vector of dimension m. As A has m rows, b also has dimension m. Here, the inequality $Ax \leq b$ represents that the ith entry of Ax is less than or equal to the ith entry of b for every $i \in [m]$. In general, given two vectors $b^1 = (b_1^1, \ldots, b_m^1), b^2 = (b_1^2, \ldots, b_m^2) \in \mathbb{R}^m$, $b^1 \leq b^2$ means that $b_i^1 \leq b_i^2$ for every $i \in [m]$. Then, what is the ith entry of Ax? It is equal to $a_i^\top x$ where $a_i^\top$ is the ith row of the matrix A. Therefore, comparing the ith entry of Ax and that of b gives us the inequality $a_i^\top x \leq b_i$. Recall that $Ax \leq b$ can be rewritten as

$$Ax = \begin{bmatrix} a_1^\top \\ a_2^\top \\ \vdots \\ a_m^\top \end{bmatrix} x = \begin{bmatrix} a_1^\top x \\ a_2^\top x \\ \vdots \\ a_m^\top x \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

We say that $Ax \leq b$ is a **system of linear inequalities**. In fact, we may have constraints of the form

$$Ax \geq b \quad \text{or} \quad Ax = b.$$

Hence, we may have a linear program of the following form

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & A^1 x \leq b^1, \\ & A^2 x \geq b^2, \\ & A^3 x = b^3, \\ & x \in \mathbb{R}^d \end{aligned} \tag{5.2}$$

where $A^1 x \leq b^1$, $A^2 x \geq b^2$, and $A^3 x = b^3$ have m_1, m_2, and m_3 constraints, respectively. Here, we may call $Ax = b$ a **system of linear equalities**. We say that a constraint in a system of the form $A^1 x \leq b^1$ or $A^2 x \geq b^2$ is **(linear) inequality constraint** and that a constraint in a system of the form $A^3 x = b^3$ a **(linear) equality constraint**.

2.1 Equality constraints to inequality constraints

In fact, we may convert (5.2) to a linear program of the form (5.1) that consists of inequality constraints only. Note that $A'x = b'$ holds if and only if

$$A'x \geq b' \quad \text{and} \quad A'x \leq b'.$$

Hence, (5.2) is equivalent to

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & A^1 x \leq b^1, \\ & A^3 x \leq b^3, \\ & A^2 x \geq b^2, \\ & A^3 x \geq b^3, \\ & x \in \mathbb{R}^d \end{aligned}$$

Here, in the last two sets of linear inequality constraints, the inequality direction is reversed. To take this into account, we observe that $A'x \geq b'$ is equivalent to

$$-A'x \leq -b'$$

obtained after multiplying $A'x \geq b'$ by -1 on both sides. Then we deduce that

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & A^1 x \leq b^1, \\ & A^3 x \leq b^3, \\ & -A^2 x \leq b^2, \\ & -A^3 x \leq b^3, \\ & x \in \mathbb{R}^d \end{aligned} \tag{5.3}$$

is an equivalent linear program of (5.2). Here, taking

$$A = \begin{bmatrix} A^1 \\ -A^2 \\ A^3 \\ -A^3 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} b^1 \\ -b^2 \\ b^3 \\ -b^3 \end{bmatrix},$$

(5.3) reduces to (5.1).

2.2 Inequality constraints to equality constraints

Given a system of linear inequality constraints $Ax \leq b$, we can convert it to a set of linear equality constraints.

Lemma 5.1. *Let $a_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$. Then $a_i^\top x \leq b_i$ if and only if*

$$\exists s_i \in \mathbb{R} \text{ such that } s_i \geq 0 \text{ and } a_i^\top x + s_i = b_i.$$

Proof. First, assume that $a_i^\top x \le b_i$. Then

$$a_i^\top x + (b_i - a_i^\top x) = b_i.$$

Hence, we may set $s_i = b_i - a_i^\top x$. Then $s_i \ge 0$ because $a_i^\top x \le b_i$, and moreover, $a_i^\top x + s_i = b_i$ by definition. Next assume that there exists some $s_i \ge 0$ such that $a_i^\top x + s_i = b_i$. Then

$$a_i^\top x = b_i - s_i \le b_i,$$

and therefore, $a_i^\top x \le b_i$ holds. □

By this lemma, it follows that

$$Ax \le b \quad \leftrightarrow \quad \exists s \in \mathbb{R}^m \text{ such that } s \ge 0 \text{ and } Ax + s = b.$$

Therefore, the optimization problem (5.1) is equivalent to

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax + s = b, \\ & s \ge 0, \\ & x \in \mathbb{R}^d, \ s \in \mathbb{R}^m. \end{aligned}$$

Hence, subject to adding some **nonnegativity constraints**, we can convert inequality constraints into equality constraints. Similarly, we can argue that

$$Ax \ge b \quad \leftrightarrow \quad \exists s \in \mathbb{R}^m \text{ such that } s \ge 0 \text{ and } Ax - s = b.$$

2.3 Free variables to nonnegative variables

In the optimization problem (5.1), the variables x may have some negative components unless the system $Ax \le b$ contains nonnegativity constraints $x \ge 0$. In that case, we say that the variables x are **free variables**. In fact, we may come up with an equivalent optimization problem where all variables are restricted to be nonnegative.

Lemma 5.2. *Note that $x_j \in \mathbb{R}$ if and only if*

$$\exists x_j^+, x_j^- \text{ such that } x_j^+, x_j^- \ge 0 \text{ and } x_j = x_j^+ - x_j^-.$$

Proof. If $x_j \ge 0$, then we set $x_j^+ = x_j$ and $x_j^- = 0$. If $x_j < 0$, then we set $x_j^+ = 0$ and $x_j^- = -x_j$. □

This lemma implies that (5.1) is equivalent to

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax \le b, \\ & x = x^+ - x^-, \\ & x^+, \ x^- \ge 0, \\ & x, \ x^+, \ x^- \in \mathbb{R}^d \end{aligned}$$

which is equivalent to

$$\begin{aligned} \min \quad & c^\top x^+ - c^\top x^- \\ \text{s.t.} \quad & Ax^+ - Ax^- \le b, \\ & x^+, \ x^- \ge 0, \\ & x^+, \ x^- \in \mathbb{R}^d. \end{aligned}$$

2.4 Standard form

The following is a linear program in **standard form**

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax = b, \\ & x \geq 0, \\ & x \in \mathbb{R}^d. \end{aligned} \tag{5.4}$$

A linear program in standard form consists of linear equality constraints and variables that are nonnegative. Remember that the linear program (5.1) has linear inequality constraints and the variables are not necessarily nonnegative. We will see that we can convert (5.1) into a linear program in standard form. In fact, we will show that the linear program (5.2) can be equivalently written in standard form.

We will combine the idea of adding slack variables to convert inequality constraints to equality constraints and the technique of replacing each free variable by the difference of two nonnegative variables.

Note that in (5.2), the system of linear inequalities $A^1 x \leq b^1$ is equivalent to

$$\exists s^1 \in \mathbb{R}^{m_1} \text{ such that } s^1 \geq 0 \text{ and } A^1 x + s^1 = b^1.$$

Moreover, the system of linear inequalities $A^2 x \geq b^2$ is equivalent to

$$\exists s^2 \in \mathbb{R}^{m_2} \text{ such that } s^2 \geq 0 \text{ and } A^2 x - s^2 = b^2.$$

Therefore, (5.2) can be rewritten as

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & A^1 x + s^1 = b^1, \\ & A^2 x - s^2 = b^2, \\ & A^3 x = b^3, \\ & s^1 \geq 0, \ s^2 \geq 0, \\ & x \in \mathbb{R}^d, \ s^1 \in \mathbb{R}^{m_1}, \ s^2 \in \mathbb{R}^{m_2}. \end{aligned} \tag{5.5}$$

Here, variables s^1 and s^2 are nonnegative, but x are free variables. Then, we can replace x by $x = x^+ - x^-$ where $x^+, x^- \in \mathbb{R}^d$ and $x^+, x^- \geq 0$. As a result,

$$\begin{aligned} \min \quad & c^\top x^+ - c^\top x^- \\ \text{s.t.} \quad & A^1 x^+ - A^1 x^- + s^1 = b^1, \\ & A^2 x^+ - A^2 x^- - s^2 = b^2, \\ & A^3 x^+ - A^3 x^- = b^3, \\ & x^+ \geq 0, \ x^- \geq 0, \ s^1 \geq 0, \ s^2 \geq 0, \\ & x^+, \ x^- \in \mathbb{R}^d, \ s^1 \in \mathbb{R}^{m_1}, \ s^2 \in \mathbb{R}^{m_2}. \end{aligned} \tag{5.6}$$

Now (5.6) is in standard form. We may express (5.6) in matrix form as follows.

$$\begin{aligned} \min \quad & \begin{bmatrix} c^\top & -c^\top & 0^\top & 0^\top \end{bmatrix} \begin{bmatrix} x^+ \\ x^- \\ s^1 \\ s^2 \end{bmatrix} \\ \text{s.t.} \quad & \begin{bmatrix} A^1 & -A^1 & I_{m_1} & 0 \\ A^2 & -A^2 & 0 & -I_{m_2} \\ A^3 & -A^3 & 0 & 0 \end{bmatrix} \begin{bmatrix} x^+ \\ x^- \\ s^1 \\ s^2 \end{bmatrix} = \begin{bmatrix} b^1 \\ b^2 \\ b^3 \end{bmatrix} \\ & x^+ \geq 0,\ x^- \geq 0,\ s^1 \geq 0,\ s^2 \geq 0, \\ & x^+,\ x^- \in \mathbb{R}^d,\ s^1 \in \mathbb{R}^{m_1},\ s^2 \in \mathbb{R}^{m_2} \end{aligned} \tag{5.7}$$

where I_{m_1} is the $m_1 \times m_1$ identity matrix and I_{m_2} is the $m_2 \times m_2$ identity matrix.

3 History of linear programming

Linear programming was first devised by Kantorovich in 1939 [Kan39]. Then LP was used to model problems in military operations research duing World War II, and during that time, Dantzig was part of Project SCOOP (Scientific Computation of Optimum Programs) arranged for Pentagon [CET16]. Dantzig developed the famous “Simplex method” for solving linear programs[1]. The following quote is from his note on the method written in 1985 [Dan85].

> <u>Origins of the Simplex Method, Summer 1947</u>
>
> The first idea that would occur to anyone as a technique for solving a linear program, aside from the obvious one of moving through the interior of the convex set, is that of moving from one vertex to the next along edges of the polyhedral set. I discarded this idea immediately as impractical in higher dimensional spaces. It seemed intuitively obvious that there would be far too many vertices and edges to wander over in the general case for such a method to be efficient.
>
> **When Hurwicz came to visit me at the Pentagon in the summer of 1947, I told him how I had discarded this vertex-edge approach as intuitively inefficient for solving LP. I suggested Instead that we study the problem in the geometry of columns rather than the usual one of the rows – column geometry incidently was the one I had used in my Ph.D. thesis on the Neyman-Pearson Lemma. We dubbed the new method “climbing the bean pole.” It looked to me efficient.**
>
> I felt sufficiently confident in this special case of what later became known as the simplex method that I proceeded to modify it so that it would work for linear programs without a convexifying row. I also developed a variant for getting a starting feasible solution called <u>Phase I</u>. It was then that I discovered that the method was really the previously discarded vertex-edge procedure in disguise (except for an added criterion for selecting the edge on which to move). **Apparently, in one geometry the simplex method looks efficient while in another it appeared to be very inefficient! Thus the <u>simplex method</u> was born in August 1947.**

[1] I was not able to find a specific document announcing the result in 1947.

The simplex method works really well in practice! However, Klee and Minty found a pathological instance for the simplex method, requiring exponential time to solve if the method is used [KM72]. Later, Khachiyan announced in 1979 that he proved that the Ellipsoid method, proposed by Naum Z. Shor [Sho77] (also Yudin and Nemirovski [YN76]), solves linear programming in (weakly) polynomial time! [Kha79], and the full proof was published in 1980 [Kha80]. Although this result is indeed a breakthrough in theory, it is a general perception that the method is not as practical. Later, Karmarkar proposed an interior-point algorithm, which is much faster than the ellipsoid method and is proved to run in polynomial time [Kar84]. There have been far greater improvements in linear programming, both in practice and theory, and it is still one of the central research topics in optimization.

The following is a list of recent progress on fast algorithms for linear programming.

- (Khachiyan, 1980 [Kha80]) $O(n^6)$ time ellipsoid method.
- (Vaidya, FOCS1989 [Vai89]) $O(n^{2.5})$ time implementation of Karmarkar's method.
- (Cohen, Lee and Song, STOC2019 [CLS19]) $O^*(n^\omega + n^{2.5-\alpha/2} + n^{2+1/6})$ time.
- (Jiang, Song, Weinstein, and Zhang, STOC2021 [JSWZ21]) $O^*(n^\omega + n^{2.5-\alpha/2} + n^{2+1/18})$ time.

Here, n is the number of variables, O^* hides $n^{o(1)}$ factors, ω is the exponent of matrix multiplication, and α is the dual exponent of matrix multiplication. Currently, there exist algorithms that achieve $\omega \sim 2.38$ and $\alpha \sim 0.31$, and it is believed that $\omega \simeq 2$ and $\alpha \simeq 1$.

References

[CET16] Richard W. Cottle, B. Curtis Eaves, and Mukund N. Thapa. *Dantzig, George B. (1914–2005)*, pages 1–14. Palgrave Macmillan UK, London, 2016. 3

[CLS19] Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 938–942, New York, NY, USA, 2019. Association for Computing Machinery. 3

[Dan85] George B. Dantzig. Impact of limeiar programing on computfer dvelopment. Technical Report 85-7, Department of Operations Research, Stanford University, June 1985. 3

[JSWZ21] Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. A faster algorithm for solving general lps. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 823–832, New York, NY, USA, 2021. Association for Computing Machinery. 3

[Kan39] Leonid V. Kantorovich. The mathematical method of production planning and organization. *Management Science*, 6:363–422, 1939. 3

[Kar84] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, STOC '84, page 302–311, New York, NY, USA, 1984. Association for Computing Machinery. 3

[Kha79] Leonid.G. Khachiyan. A polynomial algorithm in linear programming. *Doklady Akademii Nauk SSSR*, 244:1093–1096, 1979. 3

[Kha80] Leonid.G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980. 3

[KM72] Victor Klee and George J. Minty. How good is the simplex algorithm? In Oved Shisha, editor, *Inequalities III (Proceedings of the Third Symposium on Inequalities held at the University of California, Los Angeles, Calif., September 1–9, 1969, dedicated to the memory of Theodore S. Motzkin)*, pages 159–175, 1972. 3

[Sho77] Naum Z. Shor. Cut-off method with space extension in convex programming problems. *Cybernetics and systems analysis*, 13(1):94–96, 1977. 3

[Vai89] P.M. Vaidya. Speeding-up linear programming using fast matrix multiplication. In *30th Annual Symposium on Foundations of Computer Science*, pages 332–337, 1989. 3

[YN76] David B. Yudin and Arkadi S. Nemirovski. Evaluation of the information com- plexity of mathematical programming problems. *Ekonomika i Matematicheskie Metody*, 12:128–142, 1976. 3