Solving Linear Programs via the Simplex Method

Friday, April 25, 2025 2:51 PM

Outline:

References

1. Simplex method
2. minor issues
   feasible starting pt.
   degeneracy, 2-phase,
   self-dual, preprocessing

   } Nocedal + Wright book
3. Complexity — models of complexity
   worst case
   average case
   polynomial time

   } Nemirovski's class notes
4. Integer linear programs
   See notes #34

   } Vanderbei's book

(1) Simplex Method (G. Dantzig, '47 or '49)

Still (sometimes) state-of-the-art!

(Work in $\mathbb{R}^n$)

$i \in [m]$ $\quad A = \begin{bmatrix} -a_1^T- \\ \vdots \\ -a_m^T- \end{bmatrix}$

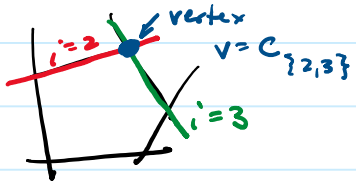
Def A **vertex** (or extreme point) of a nonempty polyhedral set $C = \{x : Ax \le b\}$ is a point v s.t. $\{v\}$ is a **face** of C (i.e. 0-dim. face)

where a **face** C_I is a set

$$C_I = \{x : a_i^T x = b_i,\ i \in I$$
$$a_i^T x < b_i,\ i \in I^c\}$$

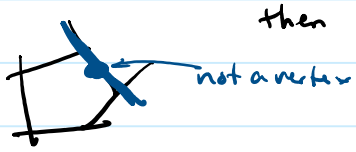
Equivalently...

algebraically... v is a vertex iff $v \in C$
and $\exists$ indices $I \subseteq [m]$ s.t. $a_i^T v = b_i \ \forall i \in I$
and $\{a_i\}_{i \in I}$ is a basis
(i.e. $|I| = n$ and $\{a_i\}_I$ lin. indep.)



geometrically... v is a vertex iff $v \in C$
and $\forall h \in \mathbb{R}^n$, if $v+h \in C$ and $v-h \in C$
then h must be 0.

(v isn't the midpoint of any segment in C)



LP (p. 2)

Friday, April 25, 2025 3:04 PM

For LP, more common to use a standard form $C = \{ x \in \mathbb{R}^n : x \geq 0, Ax = b \}$
(using slack variables if needed)

A: $m \times n$, $m < n$
Assume A has full row rank, $\text{rank}(A) = m$

$$(LP) \quad \min_{x \in \mathbb{R}^n} \; c^T x \quad \text{s.t. } Ax = b, \; x \geq 0$$

Recall...

Thm If either primal or dual of a LP has a soln (or even if feasible) then so does the other, and strong duality holds

Thm If there is a solution, there's at least one vertex solution

Simplex method checks vertices, moving to adjacent ones

In our simplex notation, call a vertex a basic feasible point

meaning $\exists$ index set $\mathcal{B} \subseteq [n]$ s.t. ("basis")

① $|\mathcal{B}| = m$

② $i \notin \mathcal{B}$ (i.e. $i \in \mathcal{N} := \mathcal{B}^c$) $\Rightarrow x_i = 0$

③ $A_{\mathcal{B}} := A(:, \mathcal{B})$ is nonsingular ("basis matrix")

This is a face, and $A_{\mathcal{B}}$ nonsingular guarantees its a single point ... so a vertex.

So $i \in \mathcal{B}$ is allowed to be nonzero *

we'll partition $[n] = \mathcal{B} \cup \mathcal{N}$

and $A = [A_{\mathcal{B}}, A_{\mathcal{N}}]$ and $x = \begin{bmatrix} x_{\mathcal{B}} \\ x_{\mathcal{N}} \end{bmatrix}$ (for exposition we assume $\mathcal{B} = \{1, \dots, m\}$, $\mathcal{N} = \{m+1, \dots, n\}$)

So $Ax = A_{\mathcal{B}} x_{\mathcal{B}} + A_{\mathcal{N}} x_{\mathcal{N}}$

Def A basis is degenerate if $\exists\, i \in \mathcal{B}$ with $x_i = 0$ (recall $i \notin \mathcal{B} \Rightarrow x_i$ has to be 0)

"The" simplex method is really a family of algs (dual, primal-dual... and "pivot rules")

We'll cover the revised simplex method

LP (p. 3)

Friday, April 25, 2025 3:18 PM

(P) $\min_x \; c^T x$ $\quad Ax=b : \lambda$, $x \geq 0 : s$

(D) $\max_{\lambda, s} \; b^T \lambda$ $\quad A^T\lambda + s = c$, $s \geq 0$

A is $m \times n$, $x \in \mathbb{R}^n$ $\quad \lambda \in \mathbb{R}^m$, $s \in \mathbb{R}^n$

write x_B, s_B, c_B to indicate subsets

KKT:

$\mathcal{L}(x, \lambda, s) = c^T x - s^T x + \lambda^T (b - Ax)$

So

① Stationarity, $\nabla_x \mathcal{L} = 0$

$0 = c - s - A^T\lambda$

② primal feas: $Ax = b$, $x \geq 0$

③ dual feas: $s \geq 0$

④ comp. slack: $x_i s_i = 0 \quad \forall i \in [n]$

Revised Simplex Method

① given a vertex x, defined by a basis $\mathcal{B} \subseteq [n]$

$Ax = b$, $Ax = A_B x_B + A_N x_N$, $x_N = 0$, $x_B = A_B^{-1} b$

② dual: choose a dual variable s, w/ $s_{\mathcal{B}} = 0$ to satisfy comp. slackness

Stationarity: $A^T\lambda + s = c$ ("pricing" step)

i.e. $\begin{bmatrix} A_B^T \\ A_N^T \end{bmatrix} \cdot \lambda + \begin{bmatrix} s_B \\ s_N \end{bmatrix} = \begin{bmatrix} c_B \\ c_N \end{bmatrix}$ ($s_B = 0$)

top equation: $A_B^T \lambda = c_B$ (A_B^T invertible)

uniquely defines λ, $\lambda = (A_B^T)^{-1} c_B$ ① (save for next pg.)

So then solve for s_N: $s_N = c_N - A_N^T \cdot \lambda$ ② (save...)

$= c_N - A_N^T A_B^{-T} c_B$

$= c_N - (A_B^{-1} A_N)^T c_B$ "reduced costs"

③a If $s_N \geq 0$, stop!

we've found a sol'n to KKT equations so we're done.

③b otherwise...

$\exists\, q \in N$ s.t. $s_q < 0$ (possibly more than 1)

Choose one such index (the "pivot") q as entering index for next vertex, i.e. we'll allow $x_q > 0$ — q enters

(temporarily) expand $\mathcal{B} \leftarrow \mathcal{B} \cup \{q\}$

Adjust $x_{\mathcal{B}}$ to maintain feasibility ($Ax = b$, $x \geq 0$):

we'll increase x_q larger and larger until

some other index p hits the boundary: $x_p = 0$. — p leaves

So $|\mathcal{B}|$ will then be back to m

if never happens, then $\min_{\substack{Ax=b \\ x \geq 0}} c^T x$ is $-\infty$

details on next pg.

LP (p. 4)

Friday, April 25, 2025 3:38 PM

x^+ is new iterate, $A x^+ = A_B x_B^+ + A_q \cdot x_q = b$ (mxm, single column, want)

and $A x^{old} = A_B x_B^{old} = b$

given a value of x_q,

then x_B^+ is determined: $x_B^+ = A_B^{-1}(b - A_q x_q^+)$

$= x_B^{old} - A_B^{-1} A_q x_q^+$

x_B^{old} strictly positive if non-degenerate

so you can see how we adjust x_q^+ until some entry of x_B^+ hits zero. (This could fail if B was degenerate)

Is this move helpful?

Look at objective: $c^T x^+ = c_B^T x_B^+ + c_q^T x_q^+$ (c_q^T unnecessary since a scalar)

recall: $\lambda = A_B^{-T} c_B$ (1)

$= c_B^T x_B^{old} - c_B^T A_B^{-1} A_q x_q^+ + c_q x_q^+$

(2) recall: $s_N = c_N - A_N^T \lambda$

i.e. $q \in N$ so $s_q = c_q - A_q^T \lambda$

$= c_B^T x_B^{old} - \lambda^T A_q x_q^+ + c_q x_q^+$

$= c_B^T x_B^{old} - (c_q - s_q) \cdot x_q^+ + c_q x_q^+$

$= c_B^T x_B^{old} + s_q x_q^+$

$s_q < 0$, $x_q^+ > 0$

So our new objective is decreased if $x_q^+ > 0$ (possible if B non-degenerate)

we chose q such that $s_q < 0$

Hence if we had several indices $s_q < 0$, one rule is to choose most negative

Thm If all vertices are nondegenerate, the simplex method will terminate (in a finite # of steps) at a basic optimal point (if one exists... else it'll detect $\min = -\infty$)

proof we always decrease objective $\Rightarrow$ can never choose a vertex twice (cycling)

and there are only finitely many vertices, so eventually we'll either try them all or find and optimal one.

Friday, April 25, 2025 3:58 PM

② Issues

1) linear algebra costs: inverting A_B.

Use update tricks: store $A_B = LU$, update efficiently

If A is sparse, a bit trickier...

2) pivoting rules ("pricing")

Dantzig's: choose q of most negative s_N, "greedy"

Not necessarily best since decrease is $s_q x_q^+$

max size of this can't be easily predicted

Partial pricing:

don't calculate all s_N in order to save computation

Heuristics

Steepest Edge ...

3) Starting: how to get a vertex to start at?

2-phase methods,

phase 1: solve $\min_{x,z} \mathbb{1}^T z$ s.t. $Ax + Ez = b$, $x, z \geq 0$

E is diagonal, $E = \text{diag}(\text{sign}(b))$

then $x = 0$, $z = |b|$ is feasible. So solve this, hopefully get $z = 0$ and x feasible

4) degenerate steps and cycling

If $\mathcal{B}$ degenerate, might not be able to take $x_q^+ > 0$ without pushing some $x_i < 0$.

Could take the step, excluding i (i.e. $\mathcal{B}^+ = \mathcal{B} \cup \{q\} \setminus \{i\}$)

... but can lead to cycling, i.e. eventually, get back to same vertex

Perturbation + lexicographic strategies

... fairly complicated, $b(\varepsilon) = b + A_B \begin{bmatrix} \varepsilon \\ \varepsilon^2 \\ \vdots \\ \varepsilon^m \end{bmatrix}$ ... see book

5) presolving

i.e., some obvious (remove redundant rows...)

some ingenious

6) dual simplex, self-dual, ... see book

Friday, April 25, 2025 4:09 PM

③ Complexity

see Nemirovskii's "Linear Optim." notes

Worst-case complexity ← nice, since independent of exact encoding, details

polynomial vs. non-polynomial

... but polynomial in what?

LP data: $\{c, b, A\}$

A) Real arithmetic encoding (aka continuous optim.)

allow $c_i, b_i, A_{ij} \in \mathbb{R}$, and unit cost for $\{+, -, \times, \sqrt{}, \div, \exp\}$...

input size = dimension = $mn + m + n$

B) Rational (aka combinatorial complexity)

assume $c_i, b_i, A_{ij} \in \mathbb{Q}$, store its binary representation

Cost of $c_i + c_j$ depends on their values (i.e. size of binary repr.)

input size = # bits to encode c, b, A

(also need to define what a "soln" is... usually need iter. bounded by $\text{poly}\log(\varepsilon^{-1})$

Ex: Solving linear eq'n $Ax = b$ is polynomial time under both models

Linear Programs

Rational/combinatorial: yes! (i.e. "weakly polynomial")

1979 L. Khachiyan ... but not via simplex, via ellipsoid method

(generalizes bisection method to $\mathbb{R}^n$, not used in practice)

builds on 1970's, Shor, Yudin, Nemirovski,

cost is $(\text{Cost}_{\text{BlackBox}}) \cdot \log(\varepsilon^{-1})$ ← friendly

80's Karmarkar, proved for IPM

Real arithmetic: Status unknown! ("Strongly polynomial"... ??)

Open problem

Simplex method not polynomial in either model

Klee-Minty: for a given pivot rule, they make an example visiting 2^n degenerate vertices

open question: is there a better pivot rule that's not combinatorially bad?

Proving polynomial Hirsch conjecture would settle it

LP (p. 7)

Friday, April 25, 2025 4:25 PM

If we fix a dimension n, then can solve LPs in strongly poly. time (or fix m and use duality), cf. Dyer, Meggido, Welzl LP chapter (and efficient in practice if $n=1$, or 2)

LP, QP, SDP...

Via IPMs, Karmarker '84 (Renegar '86, Gonzaga '86)

weakly polynomial (possibly need nondegeneracy assumptions?)

Smoothed Analysis (D. Spielman, S.-H. Teng)

Given a problem w/ data "b" (eg. "b" = (b, c, A) for LP)

and $T(b)$ = running time,

1) worst-case complexity: $\sup_{b \in B} T(b)$ eg. B bounds size of input

2) average-case: $\mathbb{E}_b\, T(b)$ Seems nice, but how to agree on what prior distribution to use?

3) Smoothed complexity: $\sup_{b \in B} \mathbb{E}_{r \sim N(0,1)} T(b + \varepsilon \cdot r)$ } mix of 1) and 2)

Paradox: worst-case simplex is 2^n (bad!) but in practice it's quite fast

Partial resolution:

Thm (Spielman, Teng '01) Simplex method is "smooth-complexity" polynomial.