11. LP Duality and the Simplex Algorithm

CPSC 535

Kevin A. Wortman

CALIFORNIA STATE UNIVERSITY FULLERTON

Recall: Standard Form

standard form with n variables and m constraints:

maximize $c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$
subject to

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_{1,n} &\leq b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_{2,n} &\leq b_2 \\ \vdots &\quad \vdots \\ a_{m,1}x_1 + a_{m,2}x_2 + \ldots + a_{m,n}x_{m,n} &\leq b_m \\ x_1, x_2, \ldots, x_n &\geq 0 \end{aligned}$$

variables: $x_1, \ldots, x_n \in \mathbb{R}$
objective function defined by coefficients $c_1, \ldots, c_n \in \mathbb{R}$
constraints defined by coefficients $a_{i,j}, b_i \in \mathbb{R}$

Recall: Standard Form Matrix Notation

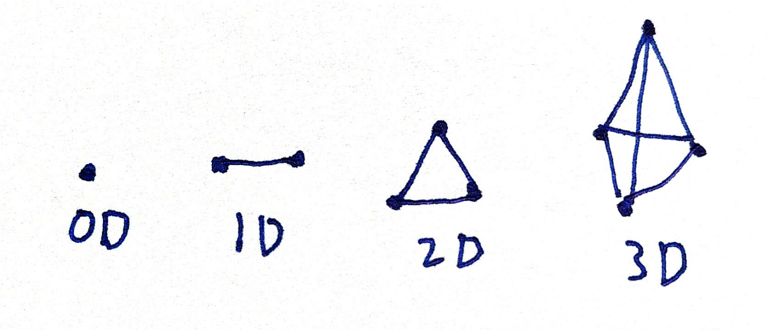
- more compact math notation
- collect:
  - variables into vector $x = \langle x_1, \ldots, x_n \rangle$
  - objective coefficients into vector $c = \langle c_1, \ldots, c_n \rangle$
  - r.h.s. of inequalities into vector $b = \langle b_1, \ldots, b_m \rangle$
  - $a_{i,j}$ coefficients into matrix A
- LP can be written in terms of dot-product and matrix-vector multiplication as (and note the transpose c^T):

maximize $c^T x$
subject to

$$\begin{aligned} Ax &\leq b \\ x &\geq 0 \end{aligned}$$

What is a Simplex?

simplex: generalization of a triangle to arbitrary dimensions



Slack Form

duality: the simplex algorithm views one LP in two ways,

1. standard form
2. *slack form*

- standard form: constraint says l.h.s $\leq$ r.h.s.
- $\Rightarrow$ the difference or "slack" between l.h.s. and r.h.s. is ≥ 0
- *slack form:* constraint says l.h.s. $+$ **slack** $=$ r.h.s.
- increasing objective $=$ decreasing slack
- introduce one new *basic variable* to represent slack in each constraint
- (pre-existing variables are *nonbasic*)
- z $=$ value of objective function
- don't bother writing "maximize" or "subject to"

Standard versus Slack Form

maximize $x_1 + 2x_2 - \frac{1}{2}x_3$
subject to

$$\begin{aligned} \frac{1}{3}x_1 + x_3 &\leq 5 \\ x_1 + x_2 + x_3 &\leq 100 \\ x_1 - x_2 &\leq -3 \\ x_1, x_2, x_3 &\geq 0 \end{aligned}$$

$$\begin{aligned} z &= x_1 + 2x_2 - \frac{1}{2}x_3 \\ x_4 &= 5 - \frac{1}{3}x_1 - x_3 \\ x_5 &= 100 - x_1 - x_2 - x_3 \\ x_6 &= -3 - x_1 + x_2 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$

basic var's: x_4, x_5, x_6
nonbasic var's: x_1, x_2, x_3

High-Level Simplex Algorithm

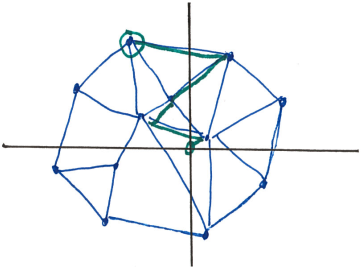
- convert standard form LP to slack form
- find a feasible (probably non-optimal) initial solution
  - intuitively: each $x_i = 0$
  - if this does not exist, return "infeasible"
- repeat:
  - choose a nonbasic variable x_i with positive coefficient in objective function (increasing x_i increases z)
    - if no such x_i exists, return solution (it's optimal)
  - increase x_i until some basic variable x_j is decreased to zero ("tighten" the slack until we're up against a constraint)
    - if none exists, return "unbounded"
  - swap roles: rewrite slack form with x_i as basic variable and x_j as nonbasic variable

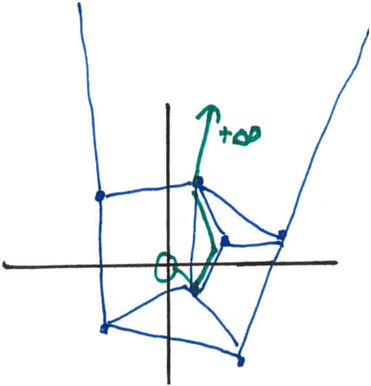
(for further details, see CLRS section 29.3)

Geometric Intuition

- a solution is a point in n-dimensional space
- intuitively, initial solution is at the origin where $x_1, \ldots, x_n = 0$
- (for further details, see CLRS section 29.5)
- each iteration "reels in" the solution to hug the intersection between two constraints
- continues until we either
  1. go "off the map" and know the LP is infeasible; or
  2. cannot improve any further $\Rightarrow$ found optimal solution
- each step moves us along the border of a *simplex*

Geometric Intuition



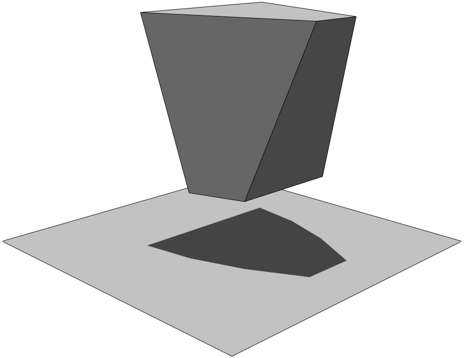


Analysis

- in LP's formulated to solve practical problems, usually
  - each of the m halfspaces intersects $O(m)$ other halfspaces
  - $\Rightarrow O(m^2)$ intersection points in the feasible region
  - $\Rightarrow$ simplex iterates $O(m^2)$ times
  - each iteration involves evaluating n-dimension obj. function
  - $\Rightarrow O(m^2 n)$ worst-case time
  - order-3 polynomial, same as max-flow
  - often faster b/c each step can "jump" pretty far
- **however,** $\exists$ feasible LP's that force simplex to take $\Omega(2^m)$ time
- *Klee-Minty cube:* $\forall d$, has $n = d$ variables, $n = d$ constraints, 2^d vertices, simplex is "tricked" into visiting all vertices
- this is a rare example of worst-case asymptotic analysis being misleading

Klee-Minty Cube

Klee-Minty Cube in 3D:



(image credit: Sophie Huiberts, CC-BY 4.0,

https://commons.wikimedia.org/wiki/File:Klee-Minty-cube-for-shadow-vertex-pivot-rule.png)

Summary

- for a standard-form LP with n variables and m constraints...
- simplex algorithm is fast in practice, technically takes $O(2^m)$ worst-case time
- Khachiyan's *ellipsoid algorithm* takes $O(n^4 W)$ time
  - seminal result, proved that sub-exponential algorithms are possible
- now have faster pseudopolynomial algorithms, e.g Vaidya's alg. takes $O((n+m)^{1.5} nW)$ time
- open questions:
  - Is there a strongly-polynomial algorithm, or is *LP* *NP*-complete?
  - Is there an algorithm that has **both** simplex' practical speed **and** provable pseudonomial runtime?