

09. Linear Programming and the Simplex Algorithm

CPSC 535 ~ Fall 2019

Kevin A. Wortman



CALIFORNIA STATE UNIVERSITY
FULLERTON

October 28, 2019



Big Ideas

- ▶ duality — same problem from different perspectives
- ▶ formulations, reductions
- ▶ visualizing high geometric dimensions

Overview

- ▶ *programming* in math involves finding some kind of optimal solution subject to mathematically-codified constraints
 - ▶ (not coding e.g. C++ programming)
- ▶ *linear programming (LP)*: optimize a linear *objective function* subject to inequalities
- ▶ very general framework
- ▶ pioneered by Soviet economist Leonid Kantorovich circa 1930s; goal was to optimize supply/demand in a communist in lieu of prices
- ▶ now used in business (*operations research*)
 - ▶ scheduling UPS deliveries, optimizing farm production, allocating investment portfolios, etc.

Computational Complexity

- ▶ many tough problems in P , including max-flow, reduce to LP
- ▶ on the border of P
- ▶ simplex algorithm technically takes $O(2^n)$ worst-case time, but is fast polynomial on most practical inputs
- ▶ we have pseudopolynomial algorithms with e.g. $O(n^{2.5}W)$ runtime and expensive constant factors
- ▶ open question whether there is a strongly polynomial LP algorithm with runtime e.g. $O(n^3)$, not a function of W

Standard Form

- ▶ *standard form*: restricted/simplified LP, easier for algorithms to solve
- ▶ later: *general form* which is more convenient for end-user formulations
- ▶ general reduces to standard with constant overhead
- ▶ similar situation to max-flow and robust max-flow
- ▶ actual solver algorithm sees a simplified standard form; reduction algorithm “frontend” accepts a generalized problem that is more convenient for end-users

Standard Form

standard form with n variables and m constraints:

maximize $c_1x_1 + c_2x_2 + \dots + c_nx_n$
subject to

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

variables: $x_1, \dots, x_n \in \mathbb{R}$

objective function defined by coefficients $c_1, \dots, c_n \in \mathbb{R}$

constraints defined by coefficients $a_{i,j}, b_i \in \mathbb{R}$

Standard Form Example

maximize $2x_1 + x_2 - \frac{1}{3}x_3$
subject to

$$x_1 + x_2 \leq 10$$

$$-x_3 \leq -2$$

$$x_1, x_2, x_3 \geq 0$$

Standard Form Matrix Notation

- ▶ more compact math notation
- ▶ collect:
 - ▶ variables into vector $x = \langle x_1, \dots, x_n \rangle$
 - ▶ objective coefficients into vector $c = \langle c_1, \dots, c_n \rangle$
 - ▶ r.h.s. of inequalities into vector $b = \langle b_1, \dots, b_m \rangle$
 - ▶ $a_{i,j}$ 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

$$Ax \leq b$$

$$x \geq 0$$

Possible Outcomes

LPs are not always solvable!

there are three outcomes:

1. **solution**: concrete values for x_1, \dots, x_n that maximize $c^T x$
(good, usually the goal)
2. **unbounded**: objective can be made arbitrarily large i.e. $+\infty$
(bad, usually means there is a bug in your LP that makes it nonsensical)
3. **infeasible**: impossible to satisfy all constraints simultaneously
(bad, usually means that either your LP is nonsensical; or your LP makes sense but meeting all your goals is impossible)

Standard-Form LP Problem

standard-form linear programming problem

input: vector $c \in \mathbb{R}^n$, vector $b \in \mathbb{R}^m$, and $m \times n$ matrix A of real numbers

output: one of

1. “unbounded”;
2. “infeasible”; or
3. “solution” with a vector $x \in \mathbb{R}^n$ maximizing the objective function

Exploring the Three Outcomes

- ▶ we will explore unbounded/infeasible/solution in 1D, then 2D
- ▶ *dimension* of an LP: #variables n
- ▶ *feasible region*: space of x vectors that satisfy all constraints
- ▶ *halfspace*: half of all geometric space,
 - ▶ 1D: one side of a point on the number line e.g. $x = 3$
 - ▶ 2D: one side of a line e.g. $y = 3x + 2$
 - ▶ 3D: one side of a plane e.g. $2x + 3y - z = 5$
- ▶ each new constraint limits the feasible region to a halfspace
- ▶ as we go, make note of
 - ▶ the shape of the feasible region
 - ▶ optimal solutions are found at extreme points (“corners”) of halfspaces
 - ▶ unbounded \Leftrightarrow feasible region extends out infinitely
 - ▶ infeasible \Leftrightarrow empty feasible region

1D Solution

maximize $2x_1$
subject to

$$x_1 \leq 4$$

$$x_1 \leq 3$$

$$x_1 \geq 0$$



- ▶ **feasible region** = intersection of all arrows = is line segment $[0, 3]$
- ▶ solution ● is $x_1 = 3$
- ▶ optimal objective function value is $2x_1 = 2(3) = 6$

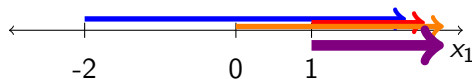
1D Unbounded

maximize $2x_1$
subject to

$$-x_1 \leq 2$$

$$-x_1 \leq -1$$

$$x_1 \geq 0$$



- ▶ **feasible region** = intersection of all arrows = open interval $[1, +\infty)$
- ▶ solution is undefined
- ▶ optimal objective function value is $2x_1 = 2(\infty) = \infty$

1D Infeasible

maximize $2x_1$
subject to

$$x_1 \leq 1$$

$$-x_1 \leq -3$$

$$x_1 \geq 0$$



- ▶ feasible region = intersection of all arrows = \emptyset
- ▶ solution is undefined
- ▶ cannot evaluate objective function

2D Solution

maximize x_2
subject to

$$\begin{aligned}\frac{1}{4}x_1 + x_2 &\leq 2 \\ -\frac{4}{5}x_1 + x_2 &\leq \frac{1}{2} \\ x_1, x_2 &\geq 0\end{aligned}$$

Sidebar: Math Definition of a Line

- ▶ recall
 - ▶ slope-intercept form $y = mx + b$
 - ▶ 2D LP constraint is $c_1x_1 + c_2x_2 \leq b$
- ▶ substitute $x_1 = x, x_2 = y$, rearrange to slope-intercept:

$$\begin{aligned}c_1x_1 + c_2x_2 &\leq b \\c_1(x) + c_2(y) &\leq b \\-(c_1x) &\quad -(c_1x) \\c_2y &\leq -c_1x + b\end{aligned}$$

if $c_2 > 0$ then

$$y \leq -\frac{c_1}{c_2}x + \frac{b}{c_2}$$

else, $c_2 < 0$, dividing by c_2 flips \leq to \geq , and

$$y \geq -\frac{c_1}{c_2}x + \frac{b}{c_2}$$

2D Solution

maximize x_2

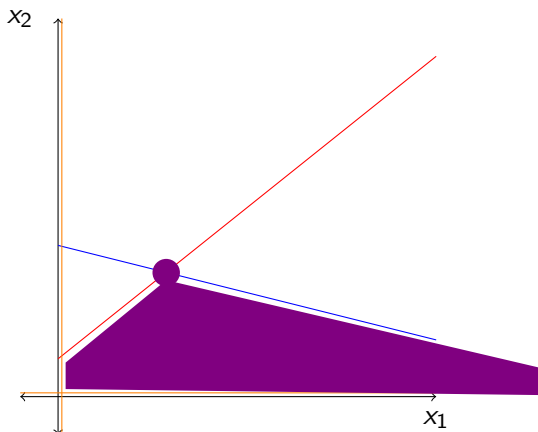
subject to

$$\frac{1}{4}x_1 + x_2 \leq 2$$

$$-\frac{4}{5}x_1 + x_2 \leq \frac{1}{2}$$

$$x_1, x_2 \geq 0$$

- ▶ feasible region is intersection of halfspaces \Leftrightarrow polygon
- ▶ optimal solution is intersection of lines at $x_1 \approx 1.43, x_2 \approx 1.64$

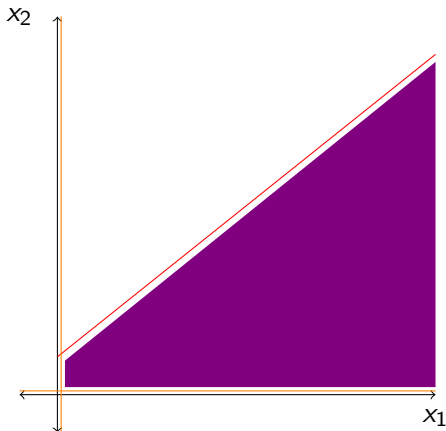


2D Unbounded

maximize x_2
subject to

$$\begin{aligned} -\frac{4}{5}x_1 + x_2 &\leq \frac{1}{2} \\ x_1, x_2 &\geq 0 \end{aligned}$$

- ▶ **feasible region** is intersection of halfspaces \Leftrightarrow some polygon sides, one infinite side
- ▶ optimal solution undefined



2D Infeasible

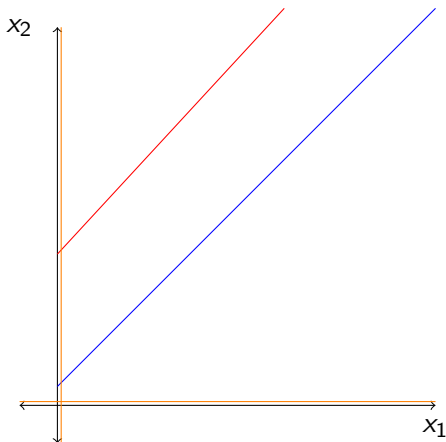
maximize x_2
subject to

$$-x_1 + x_2 \leq .25$$

$$x_1 - x_2 \leq 2$$

$$x_1, x_2 \geq 0$$

- ▶ feasible region is intersection of halfspaces \Leftrightarrow empty set
- ▶ optimal solution undefined



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

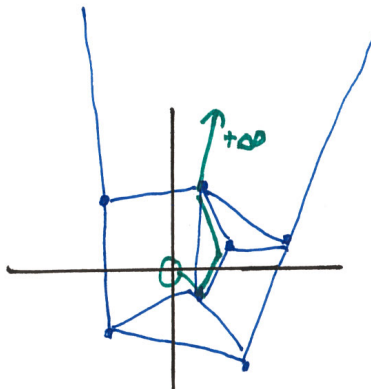
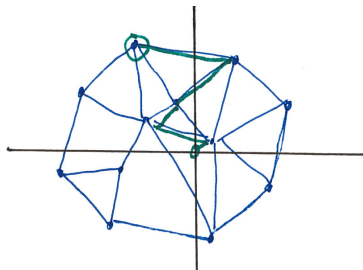
- ▶ convert standard form LP to slack form
- ▶ find a feasible (probably non-optimal) initial solution
 - ▶ if this does not exist, return “infeasible”
- ▶ repeat:
 - ▶ choose a nonbasic variable x_i with positive coefficient in objective function (increasing x_i increases z)
 - ▶ if no such x_i exists, 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 $x_1, \dots, x_n = 0$
- ▶ (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*
- ▶ simplex: n -dimensional generalization of a triangle; line segment, 2D triangle, 3D pyramid (tetrahedron), etc.

Geometric Intuition

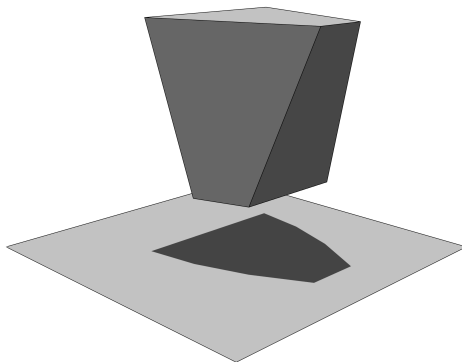


Analysis

- ▶ 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 $O((n + m)^{1.5} n W)$ 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?