Linear Programming Overview

Ronald Parr
CSCI 5951-F
Brown University

With thanks to Vince Conitzer for some content

Digression: Linear Programs

- Linear programs are ***constrained optimization problems***
- Constrained optimization problems ask us to maximize or minimize a function subject to mathematical constraints on the variables
  - Convex programs: convex objective functions, convex constraints
  - Linear programs (special case of convex programs) have linear objective functions and linear constraints
- LPs = generic language for wide range problems
- LP solvers = widely available hammers
- Entire classes and vast expertise invested in making problems look like nails

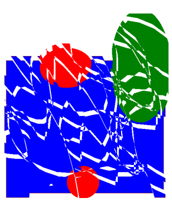
Real-World Applications

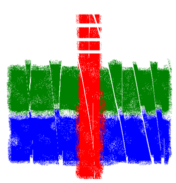
- Railroads – freight car allocation
- Agriculture – optimal mix of crops to plant
- Warfare – logistics, optimal mix of defensive assets, allocation of resources (LP techniques influenced by WWII problems)
- Networking – capacity management
- Microchips – Optimization of component placement (sort of)

Photo: Public Domain, https://commons.wikimedia.org/w/index.php?curid=17040973

Linear programs: example

- Make reproductions of 2 paintings





- Painting 1:
  - Sells for $30
  - Requires 4 units of blue, 1 green, 1 red
- Painting 2
  - Sells for $20
  - Requires 2 blue, 2 green, 1 red
- We have 16 units blue, 8 green, 5 red

maximize $3x + 2y$

subject to

$4x + 2y \leq 16$

$x + 2y \leq 8$

$x + y \leq 5$

$x \geq 0$

$y \geq 0$

Solving the linear program graphically

maximize 3x + 2y

subject to

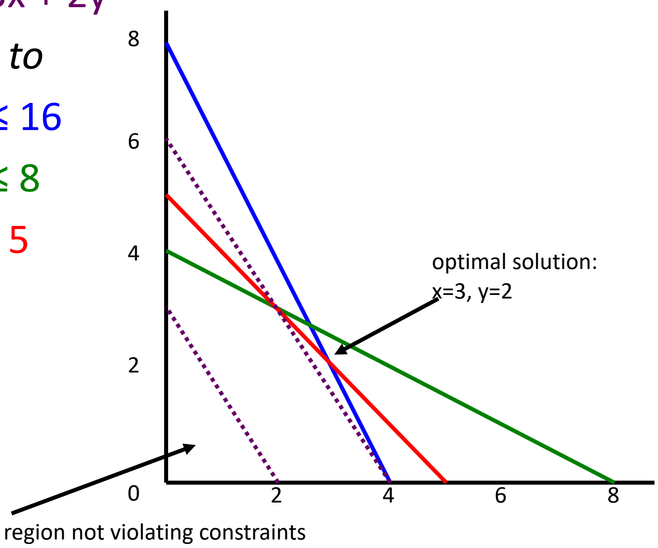
4x + 2y ≤ 16

x + 2y ≤ 8

x + y ≤ 5

x ≥ 0

y ≥ 0



Linear Programs in General

- Linear constraints, linear objective function
  - Maximize (minimize): $c^T x$ ⟵ Linear function of vector **x**
  - Subject to: $\mathbf{Ax} \leq \mathbf{b}$
    ↖ Matrix **A**
- Can swap maximize/minimize, ≤/≥; can add equality
- View as search: Searches space of values of **x**
- Alternatively: Search for tight constraints w/high objective function value

Linear Programs (max formulation)

$$\text{maximize}: c^T x$$
$$\text{subject to}: \mathbf{A}x \leq b$$
$$: x \geq 0$$

- Note: min formulation also possible
  - Min: $c^T x$
  - Subject to: $Ax \geq b$
- Some use equality as the canonical representation (introducing slack variables)
- LP tricks
  - Multiply by -1 to reverse inequalities
  - Can easily introduce equality constraints

Example: MDP Solved as an LP

$$V(s) = \max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a)V(s')$$

Issue: Turn the non-linear max into a collection of linear constraints

$$\forall s,a : V(s) \geq \underbrace{R(s,a) + \gamma \sum_{s'} P(s'|s,a)V(s')}$$

Optimal action has tight constraints

MINIMIZE: $\sum_s V(s)$

Duality

(not used in this lecture, but had to mention it because it's cool)

- For every LP there is an equivalent "Dual" problem
- Solution to primal can be used to reconstruct solution to dual, and vice versa
- LP duality:

$$\begin{aligned} \text{minimize} &: c^T x \\ \text{subject to} &: \mathbf{A}x = b \\ &: x \geq 0 \end{aligned} \quad \Longleftrightarrow \quad \begin{aligned} \text{maximize} &: b^T y \\ \text{subject to} &: \mathbf{A}^T y = c \\ &: y \geq 0 \end{aligned}$$

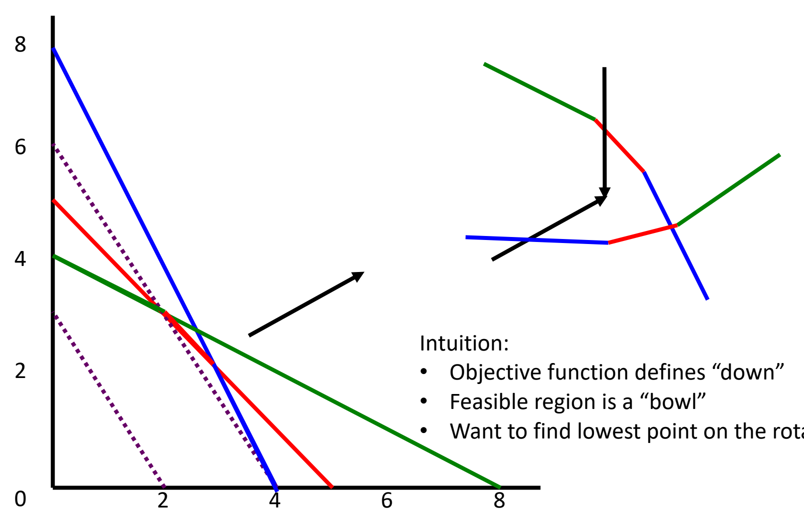
Solving linear programs (1)

- Optimal solutions always exist at vertices of the feasible region
  - Why?
  - Assume you are not at a vertex, you can always push further in direction that improves objective function
    (or at least doesn't hurt)
  - How many vertices does a kxn matrix imply?
- Dumb(est) algorithm:
  - Given n variables, k constraints
  - Check all k-choose-n = $O(k^n)$ possible vertices

Solving linear programs (2)

- Smarter algorithm (simplex)
  - Pick a vertex
  - Repeatedly hop to neighboring (one different tight constraint) vertices that improve the objective function
  - Guaranteed to find solution (no local optima)
  - May take exponential time in worst case (though rarely)
- Still smarter algorithm (interior point)
  - Move inside the interior of the feasible region, in direction that increases objective function
  - Stop when no further improvements possible
  - Tricky to get the details right, but *weakly polynomial time*

What Happens In Higher Dimensions (1)
Understanding the Feasible Region



Intuition:
- Objective function defines "down"
- Feasible region is a "bowl"
- Want to find lowest point on the rotated bowl

What Happens In Higher Dimensions (2) lines->hyperplanes

- Inequality w/2 variables -> one side of a line
- 3 variables -> one side of a plane
- k variables -> one side of hyperplane
- Physical intuition:

http://www.rubylane.com/item/623546-4085/Orrefors-x22Zenithx22-Pattern-Crystal-Bowl

Solving LPs in Practice

- Use commercial products like cplex or gurobi (there is even an Excel plug-in)
- Don't implement LP solver yourself!
- Do not use Matlab's linprog or scipy.optimize.linprog EVER. Really.
  No – REALLY!
- LP Solvers run in weakly polynomial time

Photo taken by Liane Moeller - Chris Barnes, Public Domain, https://commons.wikimedia.org/w/index.php?curid=9016956

LP Trick (one of many)

- Suppose you have a huge number of constraints, but a small number of variables (k>>n)
- Constraint generation:
  - Start with a subset of the constraints
  - Find solution to simplified LP
  - Find most violated constraint, add back to LP
  - Repeat
- Why does this work?
  - If missing constraints are unviolated, then adding them back wouldn’t change the solution
  - May terminate after adding in only a fraction of total constraints
  - No guarantees, but often helpful in practice

Linear Programming Summary

- LPs = language to express wide range of optimization problems that can be solved fairly efficiently
- Skill/art/science of modeling problems as LPs

- Nonlinear or integer versions also possible – usually lead to more accurate modeling of real-world problem, but potentially much more expensive to solve