Today:
- Linear Programming

COSC 581, Algorithms

March 27, 2014

Many of these slides are adapted from several online sources

Reading Assignments

- Today's class:
  - Chapter 29.1
- Reading assignment for next Thursday's class:
  - Chapter 29.2-3
- Announcement: Exam #2 on Tuesday, April 1
  - Will cover greedy algorithms, amortized analysis
  - HW 6-9

Linear Programming Example: A Political problem

Suppose a politician is trying to win an election

3 types of areas --- urban, suburban, rural

Certain issues --- roads, gun control, farm subsidies, gasoline tax

Politician wants to find out the minimum amount of money he/she needs to win 50,000 urban, 100,000 suburban, 25,000 rural votes.

The effects of policies on votes:

Policy	urban	suburban	rural
Build roads	-2	5	3
Gun control	8	2	-5
Farm subsidies	0	0	10
Gasoline tax	10	0	-2

How to Solve?

Define 4 variables :

x_1 is the # of thousands of dollars spent on advertising on building roads

x_2 is the # of thousands of dollars spent on advertising on gun control

x_3 is the # of thousands of dollars spent on advertising on farm subsidies

x_4 is the # of thousands of dollars spent on advertising on gasoline tax

We format this problem as a *linear program*:

Minimize:	$x_1 + x_2 + x_3 + x_4$		// Total spent
Subject to:	$-2x_1 + 8x_2 + 0x_3 + 10x_4$	≥ 50	// urban votes
	$5x_1 + 2x_2 + 0x_3 + 0x_4$	≥ 100	// suburb. votes
	$3x_1 - 5x_2 + 10x_3 - 2x_4$	≥ 25	// rural votes
	$x_1, x_2, x_3, x_4 \geq 0$		// can't have negative costs

The solution of this linear program will yield an optimal strategy for the politician.

Applications of Linear Programming

LP is a widely used Mathematical Optimization Model:

- Used frequently in management science (operations research), engineering, technology, industry, commerce, economics.
- Efficient resource allocation technique:
  - Airline transportation
  - Communication networks – e.g., optimize transmission routing
  - Factory inventory/production control
  - Fund management, stock portfolio optimization
- Can be used to approximate hard optimization problems
- . . .

History of LP

- 3000-200 BC: Egypt, Babylon, India, China, Greece: [geometry & algebra]
  - Egypt: polyhedra & pyramids.
  - India: Sulabha suutrah (Easy Solution Procedures) [2 equations, 2 unknowns]
  - China: Jiuzhang suanshu (9 Chapters on the Mathematical Art)
    [Precursor of Gauss-Jordan elimination method on linear equations]
  - Greece: Pythagoras, Euclid, Archimedes, …
- 825 AD: Persia: Muhammad ibn-Musa Alkhawrazmi (author of 2 influential books):
  "Al-Maqhaleh fi Hisab al-jabr w'almoqhabeleh" (An essay on Algebra and equations)
  "Kitab al-Jam'a wal-Tafreeq bil Hisab al-Hindi" (Book on Hindu Arithmetic).
  originated the words algebra & algorithm for solution procedures of algebraic systems.

- Fourier [1826], Motzkin [1933] [Fourier-Motzkin elimination method on linear inequalities]
- Minkowski [1896], Farkas [1902], De la Vallée Poussin [1910], von Neumann [1930's], Kantorovich [1939], Gale [1960] [LP duality theory & precursor of Simplex]

- **George Dantzig [1947]:** Simplex algorithm.
  Exponential time in the worst case, but effective in practice.
- **Leonid Khachiyan [1979]:** Ellipsoid algorithm.
  The first weakly polynomial-time LP algorithm: poly(n,d,L).
- **Narendra Karmarkar [1984]:** Interior Point Method.
  Also weakly polynomial-time. IPM variations are very well studied.
- **Megiddo-Dyer [1984]:** Prune-&-Search method.
  O(n) time if the dimension is a fixed constant. Super-exponential on dimension.

The General LP Problem

maximize $c_1 x_1 + c_2 x_2 + \cdots + c_d x_d$ ← Linear objective function

subject to:

Linear constraints (stated as inequalities)

$$a_{11} x_1 + a_{12} x_2 + \cdots + a_{1d} x_d \leq b_1$$

$$a_{21} x_1 + a_{22} x_2 + \cdots + a_{2d} x_d \leq b_2$$

$$\vdots$$

$$a_{n1} x_1 + a_{n2} x_2 + \cdots + a_{nd} x_d \leq b_n$$

Some terminology

- Objective function: value measure used to rank alternatives; either minimize or maximize this objective
- Decision variables: the quantities you can control to improve your objective function; should completely describe the set of decisions to be made
- Constraints: limitations on the values of the decision variables
- Linear program: a mathematical program in which the objective function is a linear function and the constraints are linear equalities or inequalities
- Objective value: value of objective function at a particular point
- Feasible solution: satisfies all the constraints
- Infeasible solution: doesn't satisfy the constraints
- Optimal solution: best feasible solution
- Unbounded solution: solution to LP does not have finite objective value

General Idea of LP – via another example

LP Idea: Minimize or maximize a linear objective,

Subject to linear equalities and inequalities

Example: Bubba is in a pie eating contest that lasts 1 hour. Each torte that he eats takes 2 minutes. Each apple pie that he eats takes 3 minutes. He receives 4 points for each torte and 5 points for each pie. What should Bubba eat so as to get the most points?

Step 1: *Determine the decision variables:*

Let x be the number of tortes eaten by Bubba

Let y be the number of pies eaten by Bubba

General Idea of LP – (con't)

Step 2: *Determine the objective function:*

Maximize $z = 4x + 5y$

Step 3: *Determine the constraints:*

$2x + 3y \leq 60$ // time constraints

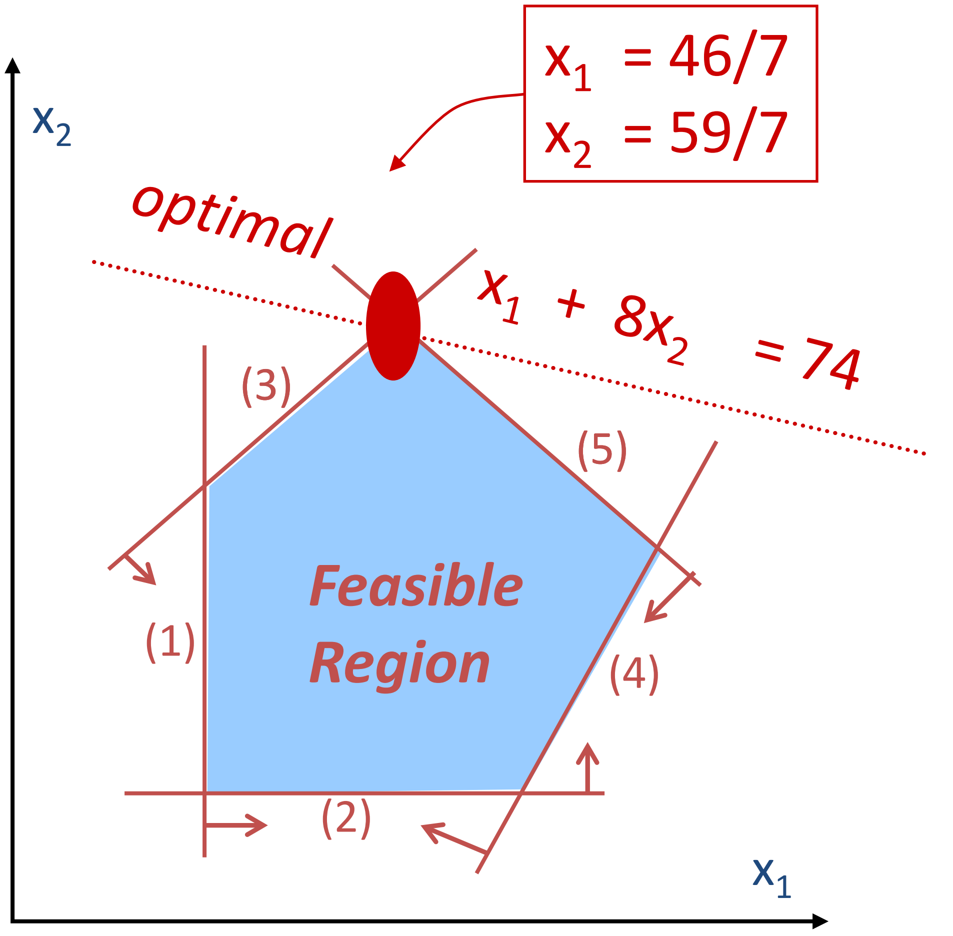
$x \geq 0$; $y \geq 0$ // non-negativity constraints – i.e., can't eat negative amounts

Visualizing LP -- Example in 2D

max $x_1 + 8x_2$

subject to:

(1) $x_1 \geq 3$
(2) $x_2 \geq 2$
(3) $-3x_1 + 4x_2 \leq 14$
(4) $4x_1 - 3x_2 \leq 25$
(5) $x_1 + x_2 \leq 15$



Each constraint is represented by a **line** and a **direction**.
Intersection of constraints is **feasible region**.

Visualizing LP -- Example in 3D

maximize z

subject to:

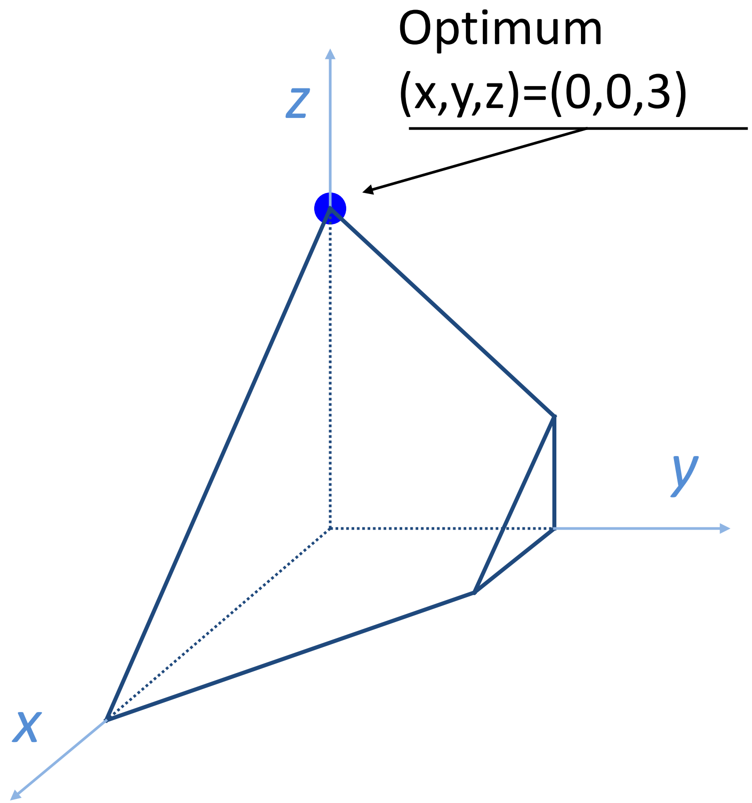
$$x + y + z \leq 3$$

$$y \leq 2$$

$$x \geq 0$$

$$y \geq 0$$

$$z \geq 0$$

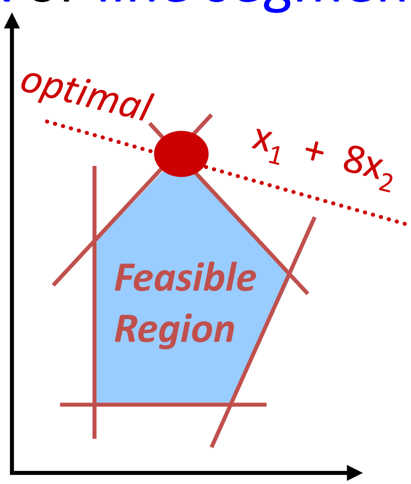


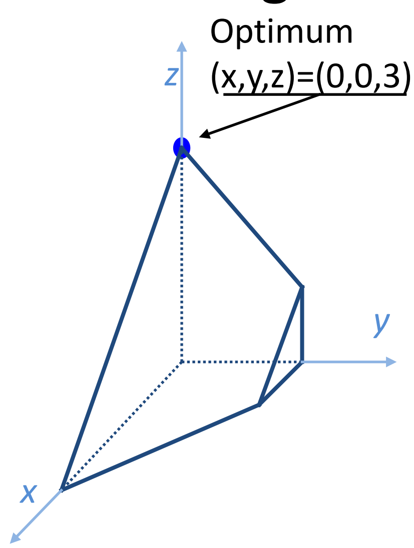
Multi-dimensional cases (beyond 3D)

- Can’t visualize beyond 3D, but same idea holds
- Each constraint defines *half-space* in n-dimensional space
- The objective function is a hyperplane
- Feasible region is area formed by intersection of half-spaces
- This region is called a *simplex*

Important Observation: Optimal Solutions are at a Vertex or Line Segment

- Intersection of objective function and feasible region is either vertex or line segment





- Feasible region is *convex* – makes optimization much easier!
- Simplex algorithm finds LP solution by:
  - Starting at some vertex
  - Moving along edge of simplex to neighbor vertex whose value is at least as large
  - Terminates when it finds local maximum
- *Convexity ensures this local maximum is globally optimal*

Recall – History of LP

- 3000-200 BC: Egypt, Babylon, India, China, Greece: [geometry & algebra]
  - Egypt: polyhedra & pyramids.
  - India: Sulabha suutrah (Easy Solution Procedures) [2 equations, 2 unknowns]
  - China: Jiuzhang suanshu (9 Chapters on the Mathematical Art)
    [Precursor of Gauss-Jordan elimination method on linear equations]
  - Greece: Pythagoras, Euclid, Archimedes, …
- 825 AD: Persia: Muhammad ibn-Musa Alkhawrazmi (author of 2 influential books):
  "Al-Maqhaleh fi Hisab al-jabr w'almoqhabeleh" (An essay on Algebra and equations)
  "Kitab al-Jam'a wal-Tafreeq bil Hisab al-Hindi" (Book on Hindu Arithmetic).
  originated the words algebra & algorithm for solution procedures of algebraic systems.

- Fourier [1826], Motzkin [1933] [Fourier-Motzkin elimination method on linear inequalities]
- Minkowski [1896], Farkas [1902], De la Vallée Poussin [1910], von Neumann [1930's], Kantorovich [1939], Gale [1960] [LP duality theory & precursor of Simplex]

- **George Dantzig [1947]:** Simplex algorithm.
  Exponential time in the worst case, but effective in practice.
- **Leonid Khachiyan [1979]:** Ellipsoid algorithm.
  The first weakly polynomial-time LP algorithm: poly(n,d,L).
- **Narendra Karmarkar [1984]:** Interior Point Method.
  Also weakly polynomial-time. IPM variations are very well studied.
- **Megiddo-Dyer [1984]:** Prune-&-Search method.
  O(n) time if the dimension is a fixed constant. Super-exponential on dimension.

Our study – while not polynomial, can be fast in practice; variants are commonly used today

In-Class exercise

Consider the following linear programming problem:

$$\text{Maximize: } x + y$$

$$\text{Subject to:}$$

$$-x + y \leq 2$$

$$2x + y \leq 6$$

$$x + 2y \leq 6$$

$$x, y \geq 0$$

What does the feasible region look like?

What is the optimal solution?

Two Canonical Forms for LP: Standard and Slack

- An LP is in standard form if it is the maximization of a linear function subject to linear inequalities

- An LP is in slack form if it is the maximization of a linear function subject to linear equalities

Standard Form

- We're given:

  n real numbers $c_1, c_2, \ldots\ c_n$
  m real numbers $b_1, b_2, \ldots\ b_m$
  mn real numbers a_{ij}, for i = 1, 2, …, m and j = 1, 2, … n

- We want to find:

  n real numbers $x_1, x_2, \ldots\ x_n$ that:

Maximize: $\sum_{j=1}^{n} c_j x_j$
Subject to:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad \text{for } i = 1, 2, \ldots, m$$

$$x_j \geq 0 \quad \text{for } j = 1, 2, \ldots, n$$

Compact Version of Standard Form

- Let: $A = (a_{ij})$ be $m \times n$ matrix
  $b = (b_i)$ be an m-vector
  $c = (c_j)$ be an n-vector
  $x = (x_j)$ be an n-vector

- Rewrite LP as:
  Maximize: $c^T x$
  Subject to:
  $$Ax \leq b$$
  $$x \geq 0$$

- Now, we can concisely specify LP in standard form as (A, b, c)

Converting LP to Standard Form

4 reasons an LP might not be in standard form:

1) Objective function might be a minimization instead of maximization
2) There might be variables w/o non-negativity constraints
3) There might be equality constraints
4) There might be inequality constraints that are " $\geq$ " instead of " $\leq$ "

We can convert any LP into an equivalent standard form

Step 1: Change min LP to max LP

- To convert a *minimization* linear program *L* into an equivalent *maximization* linear program *L'*, we simply *negate the coefficients in the objective function.*

Example:

minimize $-2x_1 + 3x_2$

subject to

$$
\begin{aligned}
x_1 + \quad x_2 &= 7 \\
x_1 - 2x_2 &\leq 4 \\
x_1 \qquad &\geq 0 \,,
\end{aligned}
$$

and we negate the coefficients of the objective function, we obtain

minimize $2x_1 - 3x_2$

subject to

$$
\begin{aligned}
x_1 + \quad x_2 &= 7 \\
x_1 - 2x_2 &\leq 4 \\
x_1 \qquad &\geq 0 .
\end{aligned}
$$

Step 2: Dealing with missing non-negativity constraints

- Suppose that some variable x_j does not have a non-negativity constraint.

- Then:
  - We replace each occurrence of x_j by $x'_j - x$

Step 3: Converting equality constraints into inequality constraints

- Suppose that a linear program has an equality constraint $f(x_1, x_2, ..., x_n) = b$.
- Since $x = y$ *if and only if* both $x \geq y$ and $x \leq y$, we can replace $f(x_1, x_2, ..., x_n) = b$ by:
  $f(x_1, x_2, ..., x_n) \leq b$ and $f(x_1, x_2, ..., x_n) \geq b$.

Step 4: Convert "≥" constraints to "≤" constraints

- We can convert the "≥" constraints to "≤" constraints by multiplying these constraints through by -1.
- That is, any inequality of the form:

$$\sum_{j=1}^{n} a_{ij} x_j \geq b_i$$

is equivalent to:

$$\sum_{j=1}^{n} -a_{ij} x_j \leq -b_i$$

Slack Form – Useful for Simplex

- In slack form, the only inequality constraints are the non-negativity constraints
  - All other constraints are equality constraints
- Let:

  $\sum_{j=1}^{n} a_{ij} x_j \leq b_i$

  be an inequality constraint

- Introduce new variable *s*, and rewrite as:

$$s = b_i - \sum_{j=1}^{n} a_{ij} x_j$$

$$s \geq 0$$

- *s* is a slack variable; it represents difference between left-hand and right-hand sides

Slack Form (con't.)

- In general, we'll use x_{n+i} (instead of s) to denote the slack variable associated with the ith inequality.

- The ith constraint is therefore:

$$x_{n+i} = b_i - \sum_{j=1}^{n} a_{ij} x_i$$

along with the non-negativity constraint $x_{n+i} \geq 0$

Example

Standard form:

Maximize $2x_1 - 3x_2 + 3x_3$

subject to:

$$x_1 + x_2 - x_3 \leq 7$$
$$-x_1 - x_2 + x_3 \leq -7$$
$$x_1 - 2x_2 + 2x_3 \leq 4$$
$$x_1, x_2, x_3 \geq 0$$

Slack form:

Maximize $2x_1 - 3x_2 + 3x_3$

subject to:

$$x_4 = 7 - x_1 - x_2 + x_3$$
$$x_5 = -7 + x_1 + x_2 - x_3$$
$$x_6 = 4 - x_1 + 2x_2 - 2x_3$$
$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0$$

About Slack Form…

Slack form:

Maximize $2x_1 - 3x_2 + 3x_3$

subject to:

$$x_4 = 7 - x_1 - x_2 + x_3$$
$$x_5 = -7 + x_1 + x_2 - x_3$$
$$x_6 = 4 - x_1 + 2x_2 - 2x_3$$
$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0$$

Basic variables – variables on left-hand side

Non-basic variables – variables on right-hand side

Concise Representation of Slack Form

- Can eliminate "maximize", "subject to", and non-negativity constraints (all are implicit)
- And, introduce *z* as value of objective function:

$$\begin{aligned} z &= 2x_1 - 3x_2 + 3x_3 \\ x_4 &= 7 - x_1 - x_2 + x_3 \\ x_5 &= -7 + x_1 + x_2 - x_3 \\ x_6 &= 4 - x_1 + 2x_2 - 2x_3 \end{aligned}$$

- Then, define slack form of LP as tuple (*N, B, A, b, c, v*)

  where *N* = indices of nonbasic variables

  B = indices of basic variables

- We can rewrite LP as:

$$z = v + \sum_{j \in N} c_j x_j$$

$$x_i = b_i - \sum_{j \in N} a_{ij} x_j \ \text{ for } i \ \in B$$

In-Class Exercise

A bank is open Monday-Friday from 9am to 5pm. From past experience, the bank knows that it needs (at least) the following number of tellers:

Time period:	9:00-10:00	10:00-11:00	11:00-12:00	12:00-1:00	1:00-2:00	2:00-3:00	3:00-4:00	4:00-5:00
Tellers required:	4	3	4	6	5	6	8	8

The bank hires two types of tellers. Full time tellers work 9-5 every day, except for 1 hour off for lunch. (The bank determines when a full time teller takes lunch hour, but it must be either 12-1 or 1-2.) Full time employees are paid $8 per hour (this includes payment for the lunch hour).

The bank can also hire part time tellers. Each part time teller must work exactly 3 consecutive hours each day, and gets paid $5 per hour. To maintain quality of service, at most 5 part time tellers can be hired.

Formulate a LP to minimize the cost of the bank to meet teller requirements.

Reading Assignments

- Reading assignment for next Thursday's class:
  - Chapter 29.2-3

- Announcement: Exam #2 on Tuesday, April 1
  - Will cover greedy algorithms, amortized analysis
  - HW 6-9