Solving Linear and Integer Programs

Robert E. Bixby

Gurobi Optimization, Inc. and Rice University

Overview

- **Linear Programming:**
  - Example and introduction to basic LP, including duality
  - Primal and dual simplex algorithms
  - Computational progress in linear programming
  - Implementing the dual simplex algorithm

Overview

- **Mixed-Integer Programming:**
  - Introduction
  - Branch and Bound
  - Computational landscape
  - Heuristics
  - Presolve and cuts
  - Parallel MIP
- **From Planning to Operations**
  - Real-time Optimization – 3 Examples

An Example

Diet Problem*

Bob wants to plan a nutritious diet, but he is on a limited budget, so he wants to spend as little money as possible. His nutritional requirements are as follows:

1. 2000 kcal
2. 55 g protein
3. 800 mg calcium

* From Linear Programming, by Vaŝek Chvátal

Diet Problem

Nutritional values

Bob is considering the following foods:

Food	Serving Size	Energy (kcal)	Protein (g)	Calcium (mg)	Price per serving
Oatmeal	28 g	110	4	2	$0.30
Chicken	100 g	205	32	12	$2.40
Eggs	2 large	160	13	54	$1.30
Whole milk	237 cc	160	8	285	$0.90
Cherry pie	170 g	420	4	22	$0.20
Pork and beans	260 g	260	14	80	$1.90

Diet Problem

Variables

We can represent the number of servings of each type of food in the diet by the variables:

x_1 servings of oatmeal

x_2 servings of chicken

x_3 servings of eggs

x_4 servings of milk

x_5 servings of cherry pie

x_6 servings of pork and beans

Diet Problem

Nutritional values

Bob is considering the following foods:

Food	Serving Size	Energy (kcal)	Protein (g)	Calcium (mg)	Price per serving	
Oatmeal	28 g	110	4	2	$0.30	x_1
Chicken	100 g	205	32	12	$2.40	x_2
Eggs	2 large	160	13	54	$1.30	x_3
Whole milk	237 cc	160	8	285	$0.90	x_4
Cherry pie	170 g	420	4	22	$2.00	x_5
Pork and beans	260 g	260	14	80	$1.90	x_6

KCAL constraint:

$$110x_1 + 205x_2 + 160x_3 + 160x_4 + 420x_5 + 260x_6 \geq 2000$$

($110x_1$ = kcals in oatmeal)

LP formulation

Minimize **Cost**

$$0.3x_1 + 2.40x_2 + 1.30x_3 + 0.90x_4 + 2.0x_5 + 1.9x_6$$

subject to: **Nutritional requirements**

$$\begin{aligned}
110x_1 + 205x_2 + 160x_3 + 160x_4 + 420x_5 + 260x_6 &\geq 2000 \\
4x_1 + 32x_2 + 13x_3 + 8x_4 + 4x_5 + 14x_6 &\geq 55 \\
2x_1 + 12x_2 + 54x_3 + 285x_4 + 22x_5 + 80x_6 &\geq 800
\end{aligned}$$

Bounds

$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0$$

Diet Problem

Solution

When we solve the preceding LP we get a solution value of $6.71, which is achieved with the following menu:

14.24 servings of oatmeal

0 servings of chicken

0 servings of eggs

2.71 servings of milk

0 servings of cherry pie

0 servings of pork and beans

Diet Problem

The Pill Salesman

A pill salesman offers Bob energy, protein, and calcium pills to fulfill his nutritional needs. We will represent the costs of each of the pills as follows:

y_1 cost (in dollars) of 1 kcal energy pill

y_2 cost (in dollars) of 1 g protein pill

y_3 cost (in dollars) of 1mg calcium pill

Diet Problem

How can I guarantee I won't make a bad deal?

Minimize **Cost**

$$0.3x_1 + 2.40x_2 + 1.30x_3 + 0.90x_4 + 2.0x_5 + 1.9x_6$$

Nutritional requirements

$$110x_1 + 205x_2 + 160x_3 + 160x_4 + 420x_5 + 260x_6 \geq 2000 \qquad \mathbf{y_1}\ \textbf{kcal}$$

$$4x_1 + 32x_2 + 13x_3 + 8x_4 + 4x_5 + 14x_6 \geq 55 \qquad \mathbf{y_2}\ \textbf{protein}$$

$$2x_1 + 12x_2 + 54x_3 + 285x_4 + 22x_5 + 80x_6 \geq 800 \qquad \mathbf{y_3}\ \textbf{calcium}$$

x_1 = servings of oatmeal: The cost of the nutrients in one serving of oatmeal shouldn't exceed the cost of just buying one serving of oatmeal:

$$110y_1 + 4y_2 + 2y_3 \leq 0.3 \quad (4\,y_2 = \text{cost of protein in oatmeal})$$

Diet Problem

The salesman's requirements

The pill salesman wants to make as much money as possible, given Bob's constraints. He knows Bob wants 2000 kcal, 55g protein, and 800 mg calcium, so his problem is as follows:

$$
\begin{array}{ll}
\text{Maximize} & 2000y_1 + 55y_2 + 800y_3 \\
\text{Subject to} & 110y_1 + 4y_2 + 2y_3 \le 0.3 \\
& 205y_1 + 32y_2 + 12y_3 \le 2.4 \\
& 160y_1 + 13y_2 + 54y_3 \le 1.3 \\
& 160y_1 + 8y_2 + 285y_3 \le 0.9 \\
& 420y_1 + 4y_2 + 22y_3 \le 2.0 \\
& 260y_1 + 14y_2 + 80y_3 \le 1.9 \\
& y_1, y_2, y_3 \ge 0
\end{array}
$$

Diet Problem

Solution

Solving this LP gives us the following pill prices:

$0.27 for 1 kcal energy pill

$0.00 for 1 g protein pill

$0.16 for 1mg calcium pill

Total cost = 0.27 (2000) + 0.16 (800) = $6.71

THE SAME AS THE LOWEST COST DIET!

Duality Theorem (John von Neumann, 1954):
These two linear programs will always give the same values

Diet Problem

Things to Remember:

The values of the "dual" variables ("shadow prices"), which are always available as part of LP solution, give the marginal value of corresponding resources.

Some Basic Theory

Linear Program – Definition

A **linear program (LP)** in **standard form** is an optimization problem of the form

$$\begin{aligned} \textit{Minimize} \quad & c^T x \\ \textit{Subject to} \quad & Ax = b \qquad\qquad (\mathrm{P}) \\ & x \geq 0 \end{aligned}$$

Where $c \in \mathbf{R}^n$, $b \in \mathbf{R}^m$, $A \in \mathbf{R}^{m \times n}$, and x is a vector of n **variables**. $c^T x$ is known as the **objective function**, $Ax=b$ as the **constraints**, and $x \geq 0$ as the **nonnegativity** conditions. b is called the **right-hand side**.

Dual Linear Program – Definition

The **dual (**or **adjoint) linear program** corresponding to (P) is the optimization problem

$$\begin{array}{rl} Maximize & b^T\pi \\ Subject\ to & A^T\pi \leq c \\ & \pi\ free \end{array} \qquad \text{(D)}$$

In this context, (P) is referred to as the **primal linear program**.

$$\left(\begin{array}{rr} \multicolumn{2}{c}{\textbf{Primal}} \\ Minimize & c^T x \\ Subject\ to & Ax = b \\ & x \geq 0 \end{array} \right)$$

Weak Duality Theorem

(von Neumann 1947)

Let x be feasible for (P) and π feasible for (D). Then

$$\boxed{\text{Maximize}} \quad b^T\pi \leq c^T x \quad \boxed{\text{Minimize}}$$

If $b^T\pi = c^T x$, then x is optimal for (P) and π is optimal for (D); moreover, if either (P) or (D) is **unbounded**, then the other problem is **infeasible**.

Proof:

$$\pi^T b \quad \underset{\boxed{Ax = b}}{=} \quad \pi^T A x \quad \underset{\boxed{\pi^T A \leq c^T \ \ \& \ \ x \geq 0}}{\leq} \quad c^T x \qquad \blacksquare$$

Solving Linear Programs

- **Three types of algorithms are available**
  - Primal simplex algorithms (Dantzig 1947)
  - Dual simplex algorithms (Lemke 1954)
    - Developed in context of game theory
  - Primal-dual log barrier algorithms
    - Interior-point algorithms (Karmarkar 1989)
    - Reference: Primal-Dual Interior Point Methods, S. Wright, 1997, SIAM

Primary focus: Dual simplex algorithms

Basic Solutions – Definition

Let B be an ordered set of m distinct indices $(B_1,\ldots,B_m)$ taken from $\{1,\ldots,n\}$. B is called a **basis** for (P) if A_B is nonsingular. The variables x_B are known as the **basic variables** and the variables x_N as the **non-basic** variables, where $N = \{1,\ldots,n\}\backslash B$. The corresponding **basic solution** $X \in \boldsymbol{R}^n$ is given by $X_N=0$ and $X_B=A_B^{-1}\, b$. B is called (**primal**) **feasible** if $X_B \geq 0$.

Note: $AX = b \;\Rightarrow\; A_B X_B + A_N X_N = b \;\Rightarrow\; A_B X_B = b \;\Rightarrow\; X_B = A_B^{-1} b$

Primal Simplex Algorithm
(Dantzig, 1947)

Input: A feasible basis B and vectors

$$X_B = A_B^{-1}b \quad \text{and} \quad D_N = c_N - A_N^T B^{-T} c_B.$$

- **Step 1:** (Pricing) If $D_N \geq 0$, stop, B is optimal; else let
$$j = argmin\{D_k : k \in N\}.$$
- **Step 2:** (FTRAN) Solve $A_B y = A_j$.
- **Step 3:** (Ratio test) If $y \leq 0$, stop, (P) is unbounded; else, let
$$i = argmin\{X_{Bk}/y_k : y_k > 0\}.$$
- **Step 4:** (BTRAN) Solve $A_B^T z = e_i$.
- **Step 5:** (Update) Compute $\alpha_N = -A_N^T z$. Let $B_i = j$. Update X_B (using y) and D_N (using α_N)

Note: x_j is called the **entering** variable and x_{Bi} the **leaving** variable. The D_N values are known as **reduced costs** – like partial derivatives of the objective function relative to the nonbasic variables.

Dual Simple Algorithm – Setup

Simplex algorithms apply to problems with constraints in equality form. We convert (D) to this form by adding the dual **slacks** *d*:

$$\begin{aligned} &\text{Maximize} && b^T\pi \\ &\text{Subject to} && A^T\pi + d = c \\ & && \pi \ \text{free}, d \geq 0 \end{aligned}$$

$$\boxed{\Leftrightarrow A^T\pi \leq c}$$

Dual Simple Algorithm – Setup

Simplex algorithms apply to problems with constraints in equality form. We convert (D) to this form by adding the dual **slacks** *d*:

$$\begin{aligned} &\text{Maximize} \quad b^T\pi \\ &\text{Subject to} \quad A^T\pi + d = c \\ &\qquad\qquad \pi \ \text{free},\ d \geq 0 \end{aligned} \longleftarrow \begin{pmatrix} A_B^T & I_B & 0 \\ A_N^T & 0 & I_N \end{pmatrix} \begin{pmatrix} \pi \\ d_B \\ d_N \end{pmatrix} = \begin{pmatrix} c_B \\ c_N \end{pmatrix}$$

Given a basis *B*, the corresponding **dual basic solution** Π, D is determined as follows:

$$D_B = 0 \ \Rightarrow \ \Pi = A_B^{-T} c_B \ \Rightarrow \ D_N = c_N - A_N^T \Pi.$$

B is **dual feasible** if $D_N \geq 0$.

An Important Fact

If X and Π, D are the respective primal and dual basic solutions determined by a basis B, then

$$\Pi^T b = c^T X.$$

Hence, by weak duality, if B is both primal and dual feasible, then X is optimal for (P) and Π is optimal for (D).

Proof: $c^T X = c_B{}^T X_B$ (since $X_N = 0$)

$= \Pi^T A_B X_B$ (since $\Pi = A_B{}^{-T} c_B$)

$= \Pi^T b$ (since $A_B X_B = b$) ∎

Dual Simplex Algorithm

(Lemke, 1954)

Input: A dual feasible basis B and vectors

$$X_B = A_B^{-1}b \quad \text{and} \quad \mathrm{D_N = c_N - A_N{}^T B^{-T} c_B}.$$

- **Step 1:** (Pricing) If $\mathrm{X_B} \geq 0$, stop, B is optimal; else let

$$i = argmin\{X_{Bk} : k \in \{1,\ldots,m\}\}.$$

- **Step 2:** (BTRAN) Solve $A_B{}^T z = e_i$. Compute $\alpha_N = -A_N{}^T z$.
- **Step 3:** (Ratio test) If $\alpha_N \leq 0$, stop, (D) is unbounded; else, let

$$j = argmin\{D_k/\alpha_k : \alpha_k > 0\}.$$

- **Step 4:** (FTRAN) Solve $\mathrm{A}_B y = A_j$.
- **Step 5:** (Update) Set $B_i = j$. Update X_B (using y) and D_N (using α_N)

Note: d_{Bi} is the **entering** variable and d_j is the **leaving** variable. (Expressed in terms of the primal: x_{Bi} is the leaving variable and x_j is the entering variable)

Simplex Algorithms

Input: A primal feasible basis B and vectors

$$X_B = A_B^{-1} b \ \ \& \ \ D_N = c_N - A_N^T A_B^{-T} c_B.$$

- **Step 1:** (Pricing) If $D_N \geq 0$, stop, B is optimal; else, let

  $$j = argmin\{D_k : k \in N\}.$$

- **Step 2:** (FTRAN) Solve $A_B y = A_j$.
- **Step 3:** (Ratio test) If $y \leq 0$, stop, (P) is unbounded; else, let

  $$i = argmin\{X_{Bk}/y_k : y_k > 0\}.$$

- **Step 4:** (BTRAN) Solve $A_B^T z = e_i$.
- **Step 5:** (Update) Compute $\alpha_N = -A_N^T z$. Let $B_i = j$. Update X_B (using y) and D_N (using α_N)

Input: A dual feasible basis B and vectors

$$X_B = A_B^{-1} b \ \ \& \ \ D_N = c_N - A_N^T A_B^{-T} c_B.$$

- **Step 1:** (Pricing) If $X_B \geq 0$, stop, B is optimal; else, let

  $$i = argmin\{X_{Bk} : k \in \{1, \ldots, m\}\}.$$

- **Step 2:** (BTRAN) Solve $A_B^T z = e_i$. Compute $\alpha_N = -A_N^T z$.
- **Step 3:** (Ratio test) If $\alpha_N \leq 0$, stop, (D) is unbounded; else, let

  $$j = argmin\{D_k/\alpha_k : \alpha_k > 0\}.$$

- **Step 4:** (FTRAN) Solve $A_B y = A_j$.
- **Step 5:** (Update) Set $B_i = j$. Update X_B (using y) and D_N (using α_N)

Correctness: Dual Simplex Algorithm

- **Termination criteria**
  - Optimality **(DONE – by "An Important Fact" !!!)**
  - Unboundedness
- **Other issues**
  - Finding starting dual feasible basis, or showing that no feasible solution exists
  - Input conditions are preserved (i.e., that B is still a feasible basis)
  - Finiteness

Summary:
What we have done and what we have to do

- **Done**
  - Defined primal and dual linear programs
  - Proved the weak duality theorem
  - Introduced the concept of a basis
  - Stated primal and dual simplex algorithms
- **To do (for dual simplex algorithm)**
  - Show correctness
  - Describe key implementation ideas
  - Motivate

Dual Unboundedness
($\Rightarrow$ primal infeasible)

- We carry out a key calculation
- As noted earlier, in an iteration of the dual

d_{Bi} enters basis
d_j leaves basis

in

$$\begin{array}{ll} \textit{Maximize} & b^T\pi \\ \textit{Subject to} & A^T\pi + d = c \\ & \pi \ \textit{free},\ d \geq 0 \end{array}$$

- **The idea:** Currently $\boldsymbol{d_{Bi} = 0}$, and $\boldsymbol{X_{Bi} < 0}$ has motivated us to increase $\boldsymbol{d_{Bi}}$ to $\boldsymbol{\theta > 0}$, leaving the other components of $\boldsymbol{d_B}$ at $\boldsymbol{0}$ (the object being to increase the objective). Letting $\underline{d}, \underline{\pi}$ be the corresponding dual solution as a function of θ, we obtain

$$\underline{\boldsymbol{d}}_B = \boldsymbol{\theta e_i} \qquad \underline{\boldsymbol{d}}_N = \boldsymbol{D_N - \theta \alpha_N} \qquad \underline{\boldsymbol{\pi}} = \boldsymbol{\Pi - \theta z},$$

where $\boldsymbol{\alpha_N}$ and $\boldsymbol{z}$ are as computed in the algorithm.

(Dual Unboundedness – cont.)

- Letting $\underline{d}, \underline{\pi}$ be the corresponding dual solution as a function of θ. Using α_N and z from dual algorithm,

$$\underline{d}_B = \theta e_i \qquad \underline{d}_N = D_N - \theta\,\alpha_N \qquad \underline{\pi} = \pi - \theta\, z.$$

- Using $\theta > 0$ and $X_{Bi} < 0$ yields

$$\begin{aligned} new\_objective &= \underline{\pi}^T b = (\pi - \theta z)^T b \\ &= \pi^T b - \theta X_{Bi} \\ &= old\_objective - \theta X_{Bi} > old\_objective \end{aligned}$$

- **Conclusion 1:** If $\alpha_N \le 0$, then $\underline{d}_N \ge 0\ \forall\ \theta > 0 \Rightarrow$ (D) is unbounded.

- **Conclusion 2:** If α_N **not**≤ 0, then

$$\underline{d}_N \ge 0 \Rightarrow \theta \le D_j/\alpha_j \ \ \forall\ \ \alpha_j > 0$$

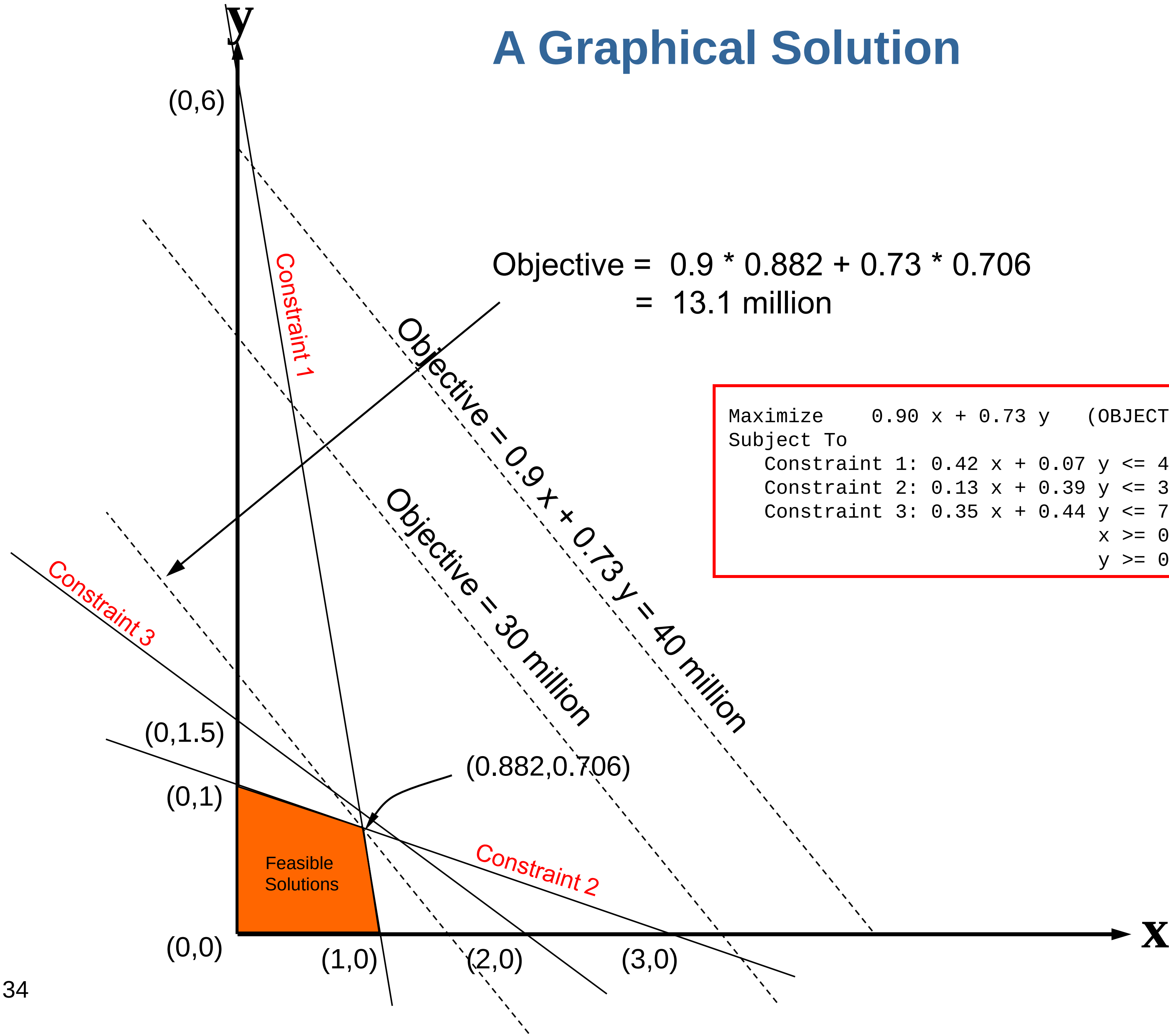
$$\Rightarrow \theta_{max} = min\{D_j/\alpha_j\!: \alpha_j > 0\}$$

(Dual Unboundedness – cont.)

- **Finiteness:** If $\boldsymbol{D_B > 0}$ for all dual feasible bases B, then the dual simplex method is finite: The dual objective strictly increases at each iteration $\Rightarrow$ no basis repeats, and there are a finite number of bases.
- There are various approaches to guaranteeing finiteness in general:
  - **Bland's Rule:** Purely combinatorial, bad in practice.
  - **CPLEX & Gurobi:** A perturbation is introduced to guarantee $\boldsymbol{D_B > 0}$.

Graphical Interpretation of Simplex Algorithms

A Graphical Solution




```
Maximize    0.90 x + 0.73 y   (OBJECTIVE)
Subject To
   Constraint 1: 0.42 x + 0.07 y <= 4200000
   Constraint 2: 0.13 x + 0.39 y <= 3900000
   Constraint 3: 0.35 x + 0.44 y <= 7000000
                               x >= 0
                               y >= 0
```

A graphical representation

We now look at a graphical representation of the simplex method as it solves the following problem:

$$\begin{array}{lrcrcrcr}
\text{Maximize} & 3x_1 & + & 2x_2 & + & 2x_3 & & \\
\text{Subject to} & x_1 & + & & & x_3 & \leq & 8 \\
& x_1 & + & x_2 & & & \leq & 7 \\
& x_1 & + & 2x_2 & & & \leq & 12 \\
& & & x_1, x_2, x_3 \geq 0 & & & &
\end{array}$$

The Primal Simplex Algorithm

Maximize $z = 3x_1 + 2x_2 + 2x_3$

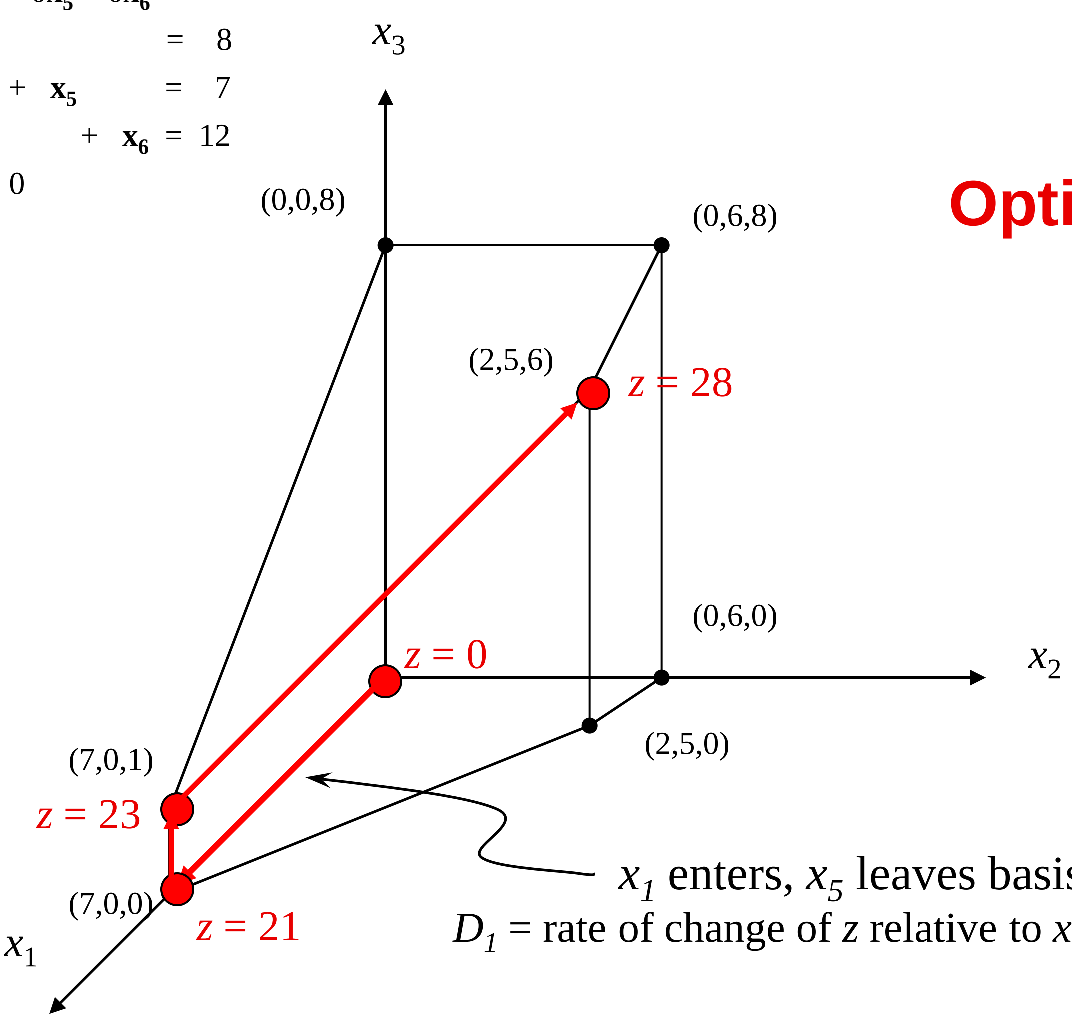
Add slacks: Initial basis B = (4,5,6)

Maximize $3x_1 + 2x_2 + 2x_3 + 0\mathbf{x_4} + 0\mathbf{x_5} + 0\mathbf{x_6}$

Subject to

$$\begin{aligned} x_1 + \quad\quad x_3 + \mathbf{x_4} \quad\quad\quad &= 8 \\ x_1 + x_2 \quad\quad\quad + \mathbf{x_5} \quad &= 7 \\ x_1 + 2x_2 \quad\quad\quad\quad + \mathbf{x_6} &= 12 \\ x_1, x_2, x_3, \mathbf{x_4}, \mathbf{x_5}, \mathbf{x_6} &\geq 0 \end{aligned}$$

Optimal!



x_1 enters, x_5 leaves basis

D_1 = rate of change of z relative to x_1 = 21/7=3

Computational History of Linear Programming

“A certain wide class of practical problems appears to be just beyond the range of modern computing machinery. These problems occur in everyday life; they run the gamut from some very simple situations that confront an individual to those connected with the national economy as a whole. Typically, these problems involve a complex of different activities in which one wishes to know which activities to emphasize in order to carry out desired objectives under known limitations.”

George B. Dantzig, 1948

The Early History

- **1947 – George Dantzig**
  - Introduced LP and recognized it as more than a conceptual tool: Computing answers important.
  - Invents simplex method for solving linear programs
  - First LP solved: Laderman, 9 cons., 77 vars., 120 MAN-DAYS.
- **1951 – First computer code for solving LPs**
- **1960 – LP commercially viable**
  - Used largely by oil companies

The Decade of the 70's

- **Interest in optimization flowered**
  - Large scale planning applications particularly popular
- **Significant difficulties emerged**
  - Building applications was very expensive and very risky
    - 3-4 year development cycles
    - Developers and application owners had to be multi-faceted experts
    - Technology just wasn't ready: LP was slow and Mixed Integer Programming was impossible.
- **Result: *Disillusionment and much of that disillusionment persists to this day.***

The Decade of the 80’s

- **Mid 80’s:**
  - There was perception was that LP software had progressed about as far as it could
- **There were several key developments**
  - IBM PC introduced in 1981
    - Brought personal computing to business
  - Relational databases developed. ERP systems introduced.
  - 1984, major theoretical breakthrough in LP (Karmarkar, Interior Point Methods, front page New York Times)
- **The last 20 years: Remarkable progress**
  - We now have three algorithms: Primal & Dual Simplex, Barrier (interior points)

Example: A Production Planning Model

401,640 constraints   1,584,000 variables

Solution time line (2.0 GHz P4)**:**

- 1988 (CPLEX 1.0):

Houston, 13 Nov 2002

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz P4)**:**

- 1988 (CPLEX 1.0): 8.0 days (Berlin, 21 Nov)

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz P4)**:**

- 1988 (CPLEX 1.0): 15.0 days (Dagstuhl, 28 Nov)

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz P4)**:**

- 1988 (CPLEX 1.0): 19.0 days (Amsterdam, 2 Dec)

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz P4)**:**

- 1988 (CPLEX 1.0): 23.0 days (Houston, 6 Dec)

Example: A Production Planning Model

401,640 constraints 1,584,000 variables

Solution time line (2.0 GHz P4)**:**

- 1988 (CPLEX 1.0): 29.8 days
- 1997 (CPLEX 5.0): 1.5 hours
- 2002 (CPLEX 8.0): 86.7 seconds
- 2003 (February): 59.1 seconds

Speedup: >43500x

Progress in LP: 1988 – 2004

- Not possible for one test to cover 15+ years: Combined several tests.
- The biggest single test:
  - **Assembled 680 real LPs**
  - Test runs: Using a time limit **(4 days per LP) two chosen methods would be compared as follows:**
    - Run method 1: Generate 680 solve times
    - Run method 2: Generate 680 solve times
    - Compute 680 ratios and form **GEOMETRIC MEAN** (not arithmetic mean!)

Progress in LP: 1988—2004

(Operations Research, Jan 2002, pp. 3—15, updated in 2004)

- **Algorithms (*machine independent*):**

  Primal *versus* best of Primal/Dual/Barrier **3,300x**

- **Machines (workstations →PCs):** **1,600x**
- **NET: Algorithm × Machine** **5,300,000x**

(2 months/5300000 ~= 1 second)

War of 1812

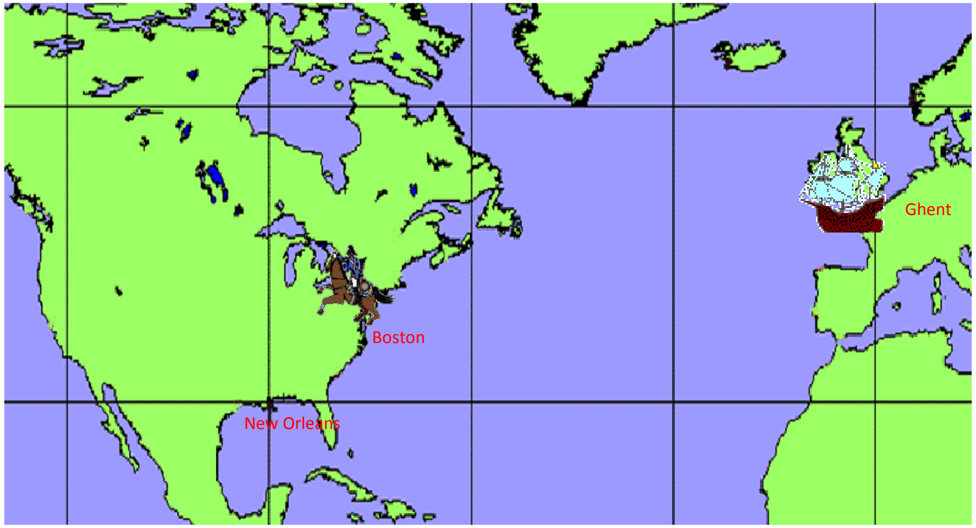
December 24, 1814: Treaty of Ghent – “ending the war”
January 8, 1815: 5300 British troops attack New Orleans

War of 1812

December 24, 1814: Treaty of Ghent – “ending the war”
January 8, 1815: 5300 British troops attack New Orleans

War of 1812

December 24, 1814: Treaty of Ghent – "ending the war"
January 8, 1815: 5300 British troops attack New Orleans



Total travel time Ghent-New Orleans:
Approximately 3 weeks

1865: Transatlantic telegraph cable completed. Communication time reduced from weeks to seconds, **a factor of ~2 million**.

Progress in LP: 1988—2004

- **Algorithm comparison**
  - Dual simplex vs. primal: Dual 2.70x faster
  - Dual simplex vs. barrier: Dual 1.06x faster
- **Where are we Today?**
  - The good news
    - "LP is a solved problem in practice"
  - But, …. a word of warning
    - 2% of MIP models are blocked by linear programming
    - Little progress in LP computation since 2004
    - LP could become a serious bottleneck in the future