Lecture 9: Linear Programming: Revised Simplex and Interior Point Methods

(a nice application of what we have learned so far)

Prof. Krishna R. Pattipati

Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna@engr.uconn.edu (860) 486-2890

ECE 6435

Adv Numerical Methods in Sci Comp

Fall 2008

October 22, 2008

Copyright ©2004 by K. Pattipati

Lecture Outline

- What is Linear Programming (LP)?
- Why do we need to solve Linear-Programming problems?
  - L_1 and L_∞ curve fitting (i.e., parameter estimation using 1-and ∞-norms)
  - Sample LP applications
    - Transportation Problems, Shortest Path Problems, Optimal Control, Diet Problem
- Methods for solving LP problems
  - Revised Simplex method
  - Ellipsoid method….not practical
  - Karmarkar's projective scaling (interior point method)
- Implementation issues of the Least-Squares subproblem of Karmarkar's method ….. More in *Linear Programming and Network Flows* course
- Comparison of Simplex and projective methods

References

1. Dimitris Bertsimas and John N. Tsisiklis, Introduction to Linear Optimization, Athena Scientific, Belmont, MA, 1997.
2. I. Adler, M. G. C. Resende, G. Vega, and N. Karmarkar, "An Implementation of Karmarkar's Algorithm for Linear Programming," Mathematical Programming, Vol. 44, 1989, pp. 297-335.
3. I. Adler, N. Karmarkar, M. G. C. Resende, and G. Vega, "Data Structures and Programming Techniques for the Implementation of Karmarkar's Algorithm," ORSA Journal on Computing, Vol. 1, No. 2, 1989.

Copyright ©2004 by K. Pattipati

What is Linear Programming?

❑ One of the most celebrated problems since 1951

- Major breakthroughs:
  - **Dantzig:** Simplex method (1947-1949)
  - **Khachian:** Ellipsoid method (1979)
    - Polynomial complexity, but not competitive with the Simplex → not practical.
  - **Karmarkar:** Projective Interior point algorithm (1984)
    - Polynomial complexity and competitive (especially for large problems)

❑ LP Problem Definition

- Given:
  - an m x n matrix A, $m < n$ or $A \in R^{mn}, m < n$ assume $rank(A) = m$
  - a column vector $\underline{b}$ with m components: $\underline{b} \in R^m$
  - a row vector $\underline{c}^T$ with n components: $\underline{c} \in R^n$

m x n $\Rightarrow$ $A\underline{x} = \underline{b}$ has infinitely many solutions $\Rightarrow$ $\underline{b} = \sum_{i=1}^{n} \underline{a}_i x_i$

consider $\underline{x}_r \in R\left(A^T\right)$, $A\underline{x}_r = \underline{b}$ $\Rightarrow$ $A\left(\underline{x}_r + \underline{x}_n\right) = \underline{b}$, where $\underline{x}_n \in N(A)$ $\Rightarrow$ $\left(\underline{x}_n : A\underline{x}_n = 0\right)$

Copyright ©2004 by K. Pattipati

Standard form of LP

We impose two restrictions on $\underline{x}$:

- We want nonnegative solutions of $A\underline{x} = \underline{b} \;\Rightarrow\; x_i \geq 0$ (or) $\underline{x} \geq 0$

$$\boxed{x \text{ such that } A\underline{x} = \underline{b} \;\&\; \underline{x} \geq 0 \text{ are said to be } \underline{\textit{feasible}}}$$

  – Among all those feasible $\{\underline{x}\}$, we want $\underline{x}^*$ such that

    $\underline{c}^T \underline{x} = c_1x_1 + c_2x_2 + \ldots c_nx_n$ is a minimum

- This leads to the so-called "standard form of LP"

$$\text{(SLP):}\quad \left.\begin{array}{l} \min \underline{c}^T \underline{x} \\ \text{s.t. } A\underline{x} = \underline{b} \\ \underline{x} \geq \underline{0} \end{array}\right)$$

convex programming problem. If a bounded solution exists, then $\underline{x}^*$ is unique $\Rightarrow$ a single minimum.

- Claim: Any LP problem can be converted into standard form.

Inequality constraints:

a) $\underline{a}_i^T \underline{x} \leq b_i \Rightarrow \left(\underline{a}_i^T \quad 1\right)\begin{pmatrix} \underline{x} \\ x_{n+1} \end{pmatrix} = b_i;\; x_{n+1} \geq 0,\;\; x_{n+1} \sim$ slack variable

– In general: $A\underline{x} \leq \underline{b} \Rightarrow A\underline{x} + \underline{y} = \underline{b} \Rightarrow \overbrace{\begin{bmatrix} A & I \end{bmatrix}}^{A_a}\begin{pmatrix} \underline{x} \\ \underline{y} \end{pmatrix} = \underline{b},\; \underline{x} \geq 0,\; \underline{y} \geq 0$

Increase number of variables by m & A_a is m by $(n+m)$ matrix.

Copyright ©2004 by K. Pattipati

How to Convert Constraints into a SLP? - 1

Inequality Constraints

b) $\underline{a}_i^T \underline{x} \geq b_i \;\Rightarrow\; \underline{a}_i^T \underline{x} - x_{n+1} = b_i, \;\; x_{n+1} \geq 0; \qquad x_{n+1} \sim$ surplus variable

$$A\underline{x} \geq \underline{b} \;\Rightarrow\; \begin{bmatrix} A & -I \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{y} \end{bmatrix} = \underline{b}, \; \underline{y} \geq \underline{0}$$

c) $d_i \leq x_i \;\Rightarrow\;$ define $\hat{x}_i = x_i - d_i \;\&\; \hat{x}_i \geq 0$

d) $d_i \geq x_i \;\Rightarrow\;$ define $\hat{x}_i = d_i - x_i \;\&\; \hat{x}_i \geq 0$

e) $d_{i1} \leq x_i \leq d_{i2} \;\Rightarrow\; 0 \leq x_i - d_{i1} \leq d_{i2} - d_{i1}$

define $\hat{x}_i = x_i - d_{i1}$ and $\hat{x}_i + \underbrace{y_i}_{\text{slack}} = d_{i2} - d_{i1}; \;\; y_i \geq 0$

f) $b_{1i} \leq \underline{a}_i^T \underline{x} \leq b_{2i} \;\Rightarrow\;$ use two slacks

$$\left.\begin{aligned} \underline{a}_i^T \underline{x} - y_{i1} &= b_{1i} \\ \underline{a}_i^T \underline{x} + y_{i2} &= b_{2i} \end{aligned}\right\} \;\; y_{i1}, y_{i2} \geq 0$$

g) $\left|\underline{a}_i^T \underline{x}\right| \leq b_i \;\Rightarrow\; -b_i \leq \underline{a}_i^T \underline{x} \leq b_i \;\Rightarrow\; \underline{a}_i^T \underline{x} - y_{i1} = -b_i; \quad \underline{a}_i^T \underline{x} + y_{i2} = b_i$

Copyright ©2004 by K. Pattipati

How to Convert Constraints into a SLP? - 2

❑ x_i is a free variable

- Define $x_i = \overline{x}_i - \hat{x}_i$ with $\overline{x}_i \geq 0$ & $\hat{x}_i \geq 0$

  a) Maximization: change $\underline{c}^T \underline{x}$ to $-\underline{c}^T \underline{x}$

  b) $\min \sum_{i=1}^{n} |x_i|$ s.t. $A\underline{x} \leq \underline{b} \Rightarrow A\underline{x} + \underline{y} = \underline{b}$; write $x_i = \overline{x}_i - \hat{x}_i$

$$\left.\begin{aligned} &\Rightarrow \min \sum_{i=1}^{n} (\overline{x}_i + \hat{x}_i) \\ &\text{s.t. } \begin{bmatrix} A & -A & I \end{bmatrix} \begin{bmatrix} \underline{\overline{x}} \\ \underline{\hat{x}} \\ \underline{y} \end{bmatrix} = \underline{b} \end{aligned}\right\}$$

The optimal solution of this problem solves the original problem.

❑ Example of LP Problems

- L_1 – curve fitting
  - Recall that given a set of scalars $(b_1, b_2, \ldots, b_m)$, the estimate that minimizes $\sum_{i=1}^{m} |x - b_i|$ is the median and that this estimate is insensitive to outliers in the data $\{b_i\}$.

Copyright ©2004 by K. Pattipati

Curve Fitting

- In the vector case, we want $\underline{x}$ such that:
  - $\min_{\underline{x}} \sum_{i=1}^{m} \left| \underline{a}_i^T \underline{x} - b_i \right| = \min_{\underline{x}} \left\| A\underline{x} - \underline{b} \right\|_1$
  - ***L_1* – curve fitting → an LP**

    write $\underline{x} = \underline{\tilde{x}} - \underline{\hat{x}}$; $\quad \left| \underline{a}_i^T \underline{x} - b_i \right| = u_i + v_i$;

    Then the LP problem is: $\min_{\underline{x},\underline{u},\underline{v}} \sum_{i=1}^{n} (u_i + v_i) = \min_{\underline{x},\underline{u},\underline{v}} \underline{e}^T (\underline{u} + \underline{v})$

    $$\text{s.t.} \quad A(\underline{\tilde{x}} - \underline{\hat{x}}) - \underline{u} + \underline{v} = \underline{b}$$

    $$\underline{\tilde{x}} \geq 0; \quad \underline{\hat{x}} \geq 0; \quad \underline{u} \geq 0; \quad \underline{v} \geq 0$$
  - ***L_∞* – curve fitting** → want $\underline{x}$ such that $\min_{\underline{x}} \max_{1 \leq i \leq m} \left| \underline{a}_i^T \underline{x} - b_i \right| = \min_{\underline{x}} \left\| A\underline{x} - \underline{b} \right\|_\infty$
  - ***L_∞* – curve fitting → an LP**
    - Let $\max_{1 \leq i \leq m} \left| \underline{a}_i^T \underline{x} - b_i \right| = w$; then the problem is equivalent to:

    $$\min_{\underline{x},w} w, \quad \text{s.t.} \; -w \leq \underline{a}_i^T \underline{x} - b_i \leq w \text{ for } i = 1, 2, \ldots, m$$

    $$\min w \quad \text{s.t.} \begin{bmatrix} A & \underline{e} \\ -A & \underline{e} \end{bmatrix} \begin{bmatrix} \underline{x} \\ w \end{bmatrix} \geq \begin{bmatrix} \underline{b} \\ -\underline{b} \end{bmatrix}$$

Copyright ©2004 by K. Pattipati

Transportation Problem - 1

- Since the number of constraints is large ($2m$) and the number of variables (n) is small, typically the dual problem with (n+1) constraints and $2m$ variables is solved instead.

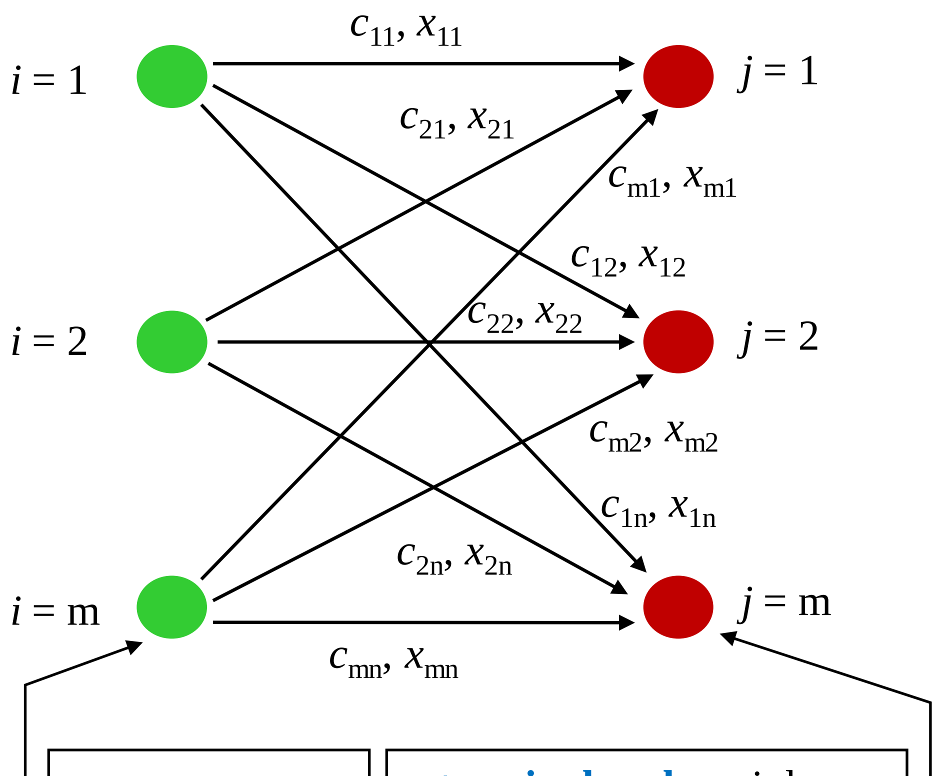
$$\max \underline{b}^T\left(\underline{\lambda}-\underline{\mu}\right)$$

$$\text{s.t. } A^T\left(\underline{\lambda}-\underline{\mu}\right)=\underline{0},\quad \underline{e}^T\left(\underline{\lambda}+\underline{\mu}\right)=1 \text{ and } \underline{\lambda}\geq \underline{0};\quad \underline{\mu}\geq \underline{0}$$

- Transportation or Hitchcock problem (special LP)
  - m sources of a commodity or a product and n destinations
  - amount of commodity to be shipped from source $i = a_i,\ 1 \leq i \leq m$
  - amount of commodity to be received at destination (sink, terminal node) $i = b_j,\ 1 \leq j \leq n$
  - shipping cost from source i to destination j per unit commodity $= c_{ij}$ dollars/unit
  - Problem: How much commodity to be shipped from source i to destination j to minimize the total cost of transportation?

Copyright ©2004 by K. Pattipati

Transportation Problem - 2



***m* sources:** supply nodes

***n* terminal nodes**: sinks, destination nodes

$$\min_{x_{ij}} \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}x_{ij}$$

$$\text{s.t.} \sum_{j=1}^{n} x_{ij} = a_i \;\; \forall\; i = 1, 2, \ldots, m$$

$$\sum_{i=1}^{m} x_{ij} = b_j \;\; \forall\; j = 1, 2, \ldots, n$$

$$\text{Also: } \sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j$$

Conservation Constraint:
mn variables & $(m+n)$ constraints

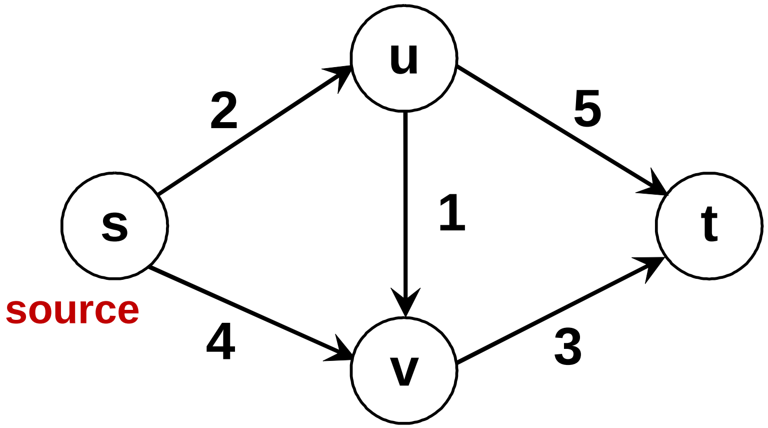
→ BIPARTITE GRAPHS: Special LP Problem

$a_i = b_i = 1 \Rightarrow$ Assignment problem or weighted bipartite matching problem.

Copyright ©2004 by K. Pattipati

Shortest Path Problem - 1

- **Shortest-path problem**
  - **We formulate it as an LP for conceptual reasons only**



- s, u, v, t are computers, edge lengths are costs of sending a message between pairs of nodes denoting computers

- **Q: What is the cheapest way to send a message from *s* to *t*?**

- Intuitively, $x_{sv} = x_{ut} = 0$, i.e., no messages are sent from s to v & from u to t.
- Shortest path $s\text{-}u\text{-}v\text{-}t \;\Rightarrow\; x_{su} = x_{uv} = x_{vt} = 1$
- Shortest path length = $2+1+3=6$

Copyright ©2004 by K. Pattipati

Shortest Path Problem - 2

LP problem formulation

- Let x_{ij} be the fraction of messages sent from i to j
  - $\min 2x_{su} + 4x_{sv} + x_{uv} + 5x_{ut} + 3x_{vt}$

    s.t. $x_{su} \geq 0;\ x_{sv} \geq 0;\ x_{uv} \geq 0;\ x_{ut} \geq 0;\ x_{vt} \geq 0$

    $x_{su} - x_{uv} - x_{ut} = 0$ (message not lost at u)

    $x_{sv} + x_{uv} - x_{vt} = 0$

    $x_{ut} + x_{vt} = 1$
- **Add all constraints → $x_{su} + x_{sv} = 1$, which it must be!!**
- **Only 3 independent constraints (although 4 nodes)**
  - In matrix notation:

$$A\underline{x} = \begin{bmatrix} 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_{su} \\ x_{sv} \\ x_{uv} \\ x_{ut} \\ x_{vt} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \underline{b}$$

  - n nodes $\Rightarrow$ $n-1$ independent equations $\Rightarrow$ Similar to Kirchoff's Laws

Copyright ©2004 by K. Pattipati

Optimal Control Problem - 1

Shortest path problem as a standard LP

$$\left.\begin{aligned} &\min \underline{e}^T \underline{x} \\ &\text{s.t. } A\underline{x} = \underline{b} \\ &\underline{x} \geq 0 \end{aligned}\right\}$$

NOTE:

- A is called the incidence matrix
- $\underline{b}$ is a special vector
- A is a unimodular matrix and so are all invertible submatrices $\tilde{A}$ of A

$\Rightarrow \det \tilde{A} = 1$ or -1

Optimal Control

- **Consider a linear time-invariant discrete-time system**

$$\underline{x}_{k+1} = A\underline{x}_k + \underline{b}u_k \;;\; u_k \sim \text{scalar for simplicity, } k = 0,1,\ldots.$$

$$\underline{x}_k = A^k \underline{x}_0 + \sum_{l=0}^{k-1} A^{k-1}\underline{b}u_l$$

– Define Terminal Error: $e_N = \underline{x}_d - x_N = \underline{x}_d - A^N \underline{x}_0 - \sum_{l=0}^{N-1} A^{N-l-1}\underline{b}u_l$

– Given $\underline{x}_0, \underline{x}_d$ & given the fact that u_k is constrained by $u_{\min} \leq u_k \leq u_{\max}$, we can formulate various versions of LP.

Copyright ©2004 by K. Pattipati

Optimal Control Problem - 2

Versions of LP

a) $\min \sum_{i=1}^{n} |e_{Ni}| = \sum_{i=1}^{n} \left| \left\{ \left(\underline{x}_d - A^N \underline{x}_0 \right)_i - \left(\sum_{l=0}^{N-1} A^{N-l-1} \underline{b} u_l \right)_i \right\} \right|$ 1-norm of error

$$= \sum_{i=1}^{n} \left| c_i + d_i^T \underline{z} \right|$$

- $d_i \sim N$ vector with components $-\left(A^{N-l-1} b\right)_i = d_{il}$
- $\underline{z} = \left(u_0, u_1, \ldots, u_{N-1}\right)^T$
- s.t. $u_{\min} \underline{1} \leq \underline{z} \leq u_{\max} \underline{1}$

• Convert to standard form via: $v_i - u_i = c_i + d_i^T \underline{z}$

$$\left.\begin{array}{ll} & \sum_{i=1}^{n} (v_i + u_i) \\ \text{s.t.} & u_{\min} \underline{1} \leq \underline{z} \leq u_{\max} \underline{1} \\ & v_i - u_i = c_i + d_i^T \underline{z} \quad 1 \leq i \leq n \end{array}\right)$$

Optimal Solutions:

$$v_i^* = \begin{cases} \underline{d}_i^T \underline{z} + c_i & \text{if } \underline{d}_i^T \underline{z} + c_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$u_i^* = \begin{cases} -\left(\underline{d}_i^T \underline{z} + c_i\right) & \text{if } \underline{d}_i^T \underline{z} + c_i < 0 \\ 0 & \text{otherwise} \end{cases}$$

- Can also include constraints on state variables

Copyright ©2004 by K. Pattipati

Optimal Control Problem - 3

❑ Versions of LP

b) $\min \max_{1\le i\le n} |e_{Ni}| = \min \max_{1\le i\le n} |c_i + d_i^T \underline{z}|$ ∞-norm of error

Define $v = \max_{1\le i\le n} |c_i + d_i^T \underline{z}| \Rightarrow \min v$

s.t. $u_{\min} \underline{1} \le \underline{z} \le u_{\max} \underline{1}, \quad v + c_i + d_i^T \underline{z} \ge 0, \quad v - c_i - d_i^T \underline{z} \ge 0$

- **Proof of equivalence for (a)**

  Suppose $v_i^*, u_i^*, \&\ z^*$ are optimal solutions. Claim: v_i^* & u_i^* can not simultaneously be non-zero.

  – If they are and $v_i^* > u_i^*$,define $\hat{v}_i = v_i^* - u_i^*$, $\hat{u}_i = 0$

  $\Rightarrow \hat{v}_i + \hat{u}_i = v_i^* - u_i^* < v_i^* + u_i^*$.... a contradiction.

  $\Rightarrow$ Only one of the two can be non-zero.

- **Proof of equivalence for (b)**

  Let z^*, v^* be optimal for revised problem, but z^* is not optimal for original problem.

  – Suppose $\hat{z}$ is optimal solution of original problem.

  – Define $v = \max |d_i^T \hat{z} + c_i| \Rightarrow$ feasible for revised problem

  $\Rightarrow v < v^* \Rightarrow$ Contradiction.

Copyright ©2004 by K. Pattipati

Diet Problem

❑ Diet Problem

- We want to find the most economical diet that meets minimum daily requirements for calories and such nutrients as proteins, calcium, iron, and vitamins.
  - We have n different food items:

    $c_j =$ cost of food item j

    $x_j =$ units of food item j (in grams) included in our economic diet
  - There are m minimum nutritional requirements

    $b_i =$ minimum daily requirement of i^{th} nutrient

    $a_{ij} =$ amount of nutrient i provided by a unit of food item j
- **The problem is an LP:**

$$\left.\begin{aligned} &\min \sum_{j=1}^{n} c_j x_j \\ &\text{s.t.} \quad \sum_{j=1}^{n} a_{ij} x_j \geq b_i; \quad i = 1, 2, \ldots, m \\ &\qquad x_j \geq 0; \quad j = 1, 2, \ldots, n \end{aligned}\right\} \Rightarrow \begin{aligned} &\min \underline{c}^T \underline{x} \\ &\text{s.t.} \quad A\underline{x} \geq \underline{b} \\ &\qquad \underline{x} \geq 0 \end{aligned}$$

Copyright ©2004 by K. Pattipati

Classes of Algorihtms for LP

Fundamental Property of LP

- Optimal solution $\underline{x}^*$ is such that $(n-m)$ of its components are zero.
- If we know the $n-m$ components that are zero, we can immediately compute the optimal solution (i.e., remaining m nonzero components) from $A\underline{x}=\underline{b}$
  - Since we don't know the zeros *a priori*, the chief task of every algorithm is to discover where they belong.

Three Classes of Algorithms for LP

- **Simplex**
- **Ellipsoid**
- **Projective Transformation (scaling) Algorithm**

Key Ideas of Simplex Algorithm

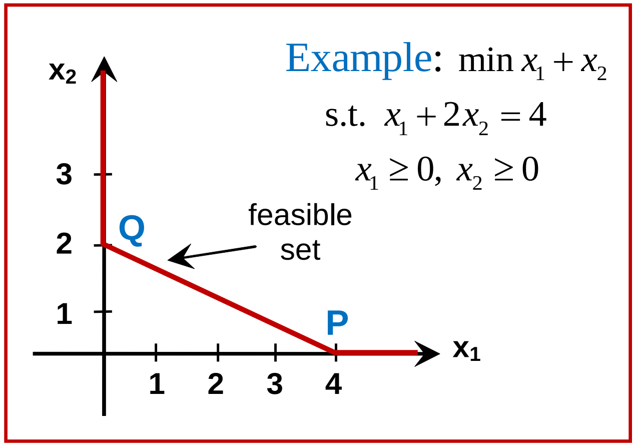
- Phase 1: Find a vector $\underline{x}$ that has $(n-m)$ zero components, with $A\underline{x}=\underline{b}$ and $\underline{x}\geq 0$. This is a feasible $\underline{x}$, not necessarily optimal.
- Phase 2: Allow one of the zero components to become positive and force one of the positive components to become zero.

Copyright ©2004 by K. Pattipati

Geometry of LP - 1

Simplex Algorithm

- Q: How to pick "entering" and "leaving" components?
- A: cost $\underline{c}^T \underline{x} \downarrow$ and $A\underline{x} = \underline{b}$, $\underline{x} \geq 0$ must be satisfied.
- **Another Key Property:** Need to look at only <u>extreme</u> (corner) points of the feasible set.



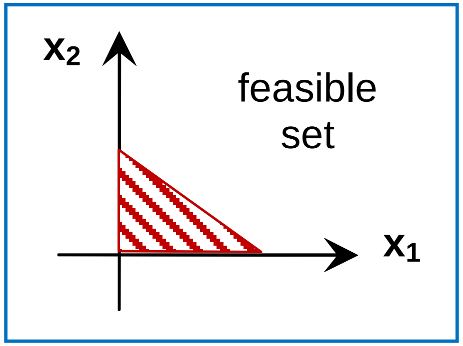
- Minimum occurs at one of the corners (vertices) of the fesible set:

  $x_1 = 0, x_2 = 2 \Rightarrow$ corner point Q
- In n-dimensions, feasible set lies in n-dimensions and so do the cost planes $\underline{c}^T \underline{x} = \text{const.}$
- Inequality constraints:

  $x_1 + 2x_2 \leq 4$, and $x_1 \geq 0,\ x_2 \geq 0$

- $\underline{x} \geq 0$ defines a positive cone in R^n.
- $\underline{a}_i^T \underline{x} \leq 0$ defines a half space on or below the plane $\underline{a}_i^T \underline{x} = 0$
- Feasible set = positive cone $\cap$ half spaces defined by $\underline{a}_i^T \underline{x} \leq b_i$

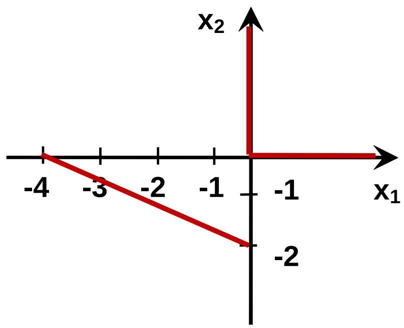
  $\Rightarrow$ polyhedron (polygon in 2 dimensions).
- Feasible set is convex: $\underline{x}_1, \underline{x}_2$ feasible $\Rightarrow \alpha \underline{x}_1 + (1-\alpha)\underline{x}_2$ is also feasible $\forall\ \alpha \in [0,1]$. Line segment is also in feasible set.



Copyright ©2004 by K. Pattipati

Geometry of LP - 2

- An LP may not have a solution



Example:

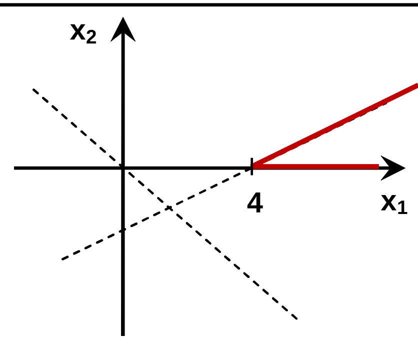
$$\min x_1 + x_2$$

$$\text{such that } x_1 + 2x_2 = -4$$

$$x_1, x_2 \geq 0$$

feasible set is empty

- An LP may have an unbounded solution



Example:

$$\min -(x_1 + x_2)$$

$$\text{such that } x_1 - 2x_2 = 4$$

$$\Rightarrow \text{ opt. } x_1, x_2 = (\infty, \infty)$$

$$\Rightarrow \text{ opt. cost value } = -\infty$$

- So, an algorithm must decide whether an optimal solution exists and find the corner where the optimum occurs.

Copyright ©2004 by K. Pattipati

Revised Simplex Algorithm - 1

Revised Simplex Algorithm

- Consider SLP: $\min \underline{z} = \underline{c}^T \underline{x}$ s.t. $A\underline{x} = \underline{b}$ and $\underline{x} \geq 0$
  - Assume $rank(A) = m.$ Then, we can partition $A = [B \mid N]$, where $B \sim m$ linearly independent columns.
  - Assume first m columns for convenience

$$[B \mid N]\begin{bmatrix} \underline{x}_B \\ --- \\ \underline{x}_N \end{bmatrix} = \underline{b} \qquad \underline{x}_B \in R^m; \quad \underline{x}_N \in R^{n-m}$$

- We know $n-m$ components of $\underline{x}$ are zero
  - If $\underline{x}_N = 0, \underline{x}_B = B^{-1}\underline{b}$ is said to be the basic solution and the columns of B form the basis
  - If, in addition, $\underline{x}_B \geq \underline{0}$, then $\underline{x}_B$ is called the basic feasible solution.
  - In terms of $\underline{x}_B$ and $\underline{x}_N$, the cost function is

$$z = \underline{c}_B^T \underline{x}_B + \underline{c}_N^T \underline{x}_N$$

  - Using $\underline{x}_B = B^{-1}\underline{b} - B^{-1}N\underline{x}_N = B^{-1}\underline{b} - B^{-1}\left(\underline{a}_{m+1}x_{m+1} + \ldots + \underline{a}_n x_n\right)$

$$\Rightarrow z = \underline{c}_B^T B^{-1}\underline{b} + \left(\underline{c}_N^T - \underline{c}_B^T B^{-1}N\right)\underline{x}_N = z_0 + \underline{p}^T \underline{x}_N$$

$$= z_0 + p_1 x_{m+1} + \ldots + p_{n-m} x_n$$

Copyright ©2004 by K. Pattipati

Revised Simplex Algorithm - 2

Revised Simplex Algorithm

- Compute $\underline{p}^T$ in two steps:

  1) Solve: $B^T \underline{\lambda} = \underline{c}_B$

  2) Compute: $\underline{p}^T = \underline{c}_N - \underline{\lambda}^T N$ or $\underline{p} = \underline{c}_N - N^T \underline{\lambda}$;

  $\underline{\lambda}$ is called vector of simplex multipliers

  – $\underline{p}$ is called the relative cost vector

  $\Rightarrow$ forms the basis fo exchanging basis variables.

  – If $\underline{p} \geq 0$, then the corner is optimal, since $\underline{p}^T \underline{x}_N = 0$ and $\underline{x} \geq \underline{0}$,

  it doesn't pay to increase $\underline{x}_N$.

  – If a component $p_k < 0$, then the cost can be decreased by increasing the corresponding component of $\underline{x}_N$, that is, $\left(x_k : m+1 \leq k \leq n\right)$.

- **Simplex method chooses one entering variable**
  - **One with the most negative p_k (or)**
  - **The first negative p_k (avoids cycling)**
- Simplex allows the component x_k to increase from zero.

Copyright ©2004 by K. Pattipati

Revised Simplex Algorithm - 3

Revised Simplex Algorithm

- Q: Which component x_1 should leave?
- A: It will be the first to reach zero. $\Rightarrow A\underline{x} = \underline{b}$ is satisfied again at the new point.
- Assume $p_k < 0$, and consider what happens when we increase x_{Nk} from zero.
  - Let $\underline{x}_B^{old}$ = initial feasible solution

    $$\underline{x}_B^{new} + B^{-1}\underline{a}_k x_{Nk} = B^{-1}\underline{b} = \underline{x}_0 = \underline{x}_B^{old}$$

    $$\Rightarrow \underline{x}_B^{new} + x_{Nk}\underline{y} = B^{-1}\underline{b} = \underline{x}_0, \text{ where } B\underline{y} = \underline{a}_k$$

  - i^{th} component of $\underline{x}_B^{new}$ will be zero when the i^{th} component of

    $\underline{y}x_{Nk} = y_i x_{Nk}$, and $\left(B^{-1}\underline{b}\right)_i = x_{0i}$ are equal. This happens when

    $\Rightarrow x_{Nk} = i^{th}$ component of $B^{-1}\underline{b}\,/\,i^{th}$ component of $\underline{y} = x_{0i} / y_i$
- So, among all y_is such that $y_i > 0$, the smallest of these ratios determines how large x_{Nk} can become.
  - If the l^{th} ratio is the smallest, then the leaving variable will be x_l.
    - At the new corner, $x_{Nk} > 0$ and $x_l = 0$.
    - $x_{Bl} \Rightarrow$ nonbasic set & column $\underline{a}_l$ joins the nonbasic matrix N.
    - $x_k \Rightarrow$ basic set & column k joins the basic matrix B.
- Thus, $\theta = \dfrac{x_{0l}}{y_l} = \min\limits_{1 \le i \le m}\left(\dfrac{x_{0i}}{y_i} : y_i > 0\right)$

Copyright ©2004 by K. Pattipati

Revised Simplex Algorithm Steps

One Iteration of Revised Simplex Algorithm

- **Step 1:** Given is the basis B such that $\underline{x}_B = B^{-1}\underline{b} \geq \underline{0}$.
- **Step 2:** Solve $B^T\underline{\lambda} = \underline{c}_B$ for the vector of simplex multipliers $\underline{\lambda}$.
- **Step 3:** Select a column $\underline{a}_k$ of N such that $\underline{p}_k = c_{Nk} - \underline{\lambda}^T\underline{a}_k < 0$. We may, for example, select the $\underline{a}_k$ which gives the largest negative values of p_k or the first k with negative p_k.
  - If $\underline{p}^T = \underline{c}_N - \underline{\lambda}^T N \geq 0$, stop $\Rightarrow$ current solution is optimal.
- **Step 4:** Solve for $\underline{y}$: $B\underline{y} = \underline{a}_k$
- **Step 5:** Find $\theta = x_{0l}/y_l = \min\left(x_{0i}/y_i\right)$ where $1 \leq i \leq m$ and $y_i > 0$.
  - Look at $x_{Bi}^{new} = x_{0i} - y_i x_{Nk}$.
  - If none of the y_is are positive, then the set of solutions to $A\underline{x} = \underline{b}$, $\underline{x} \geq 0$ is unbounded and the cost z can be made an arbitrarily large negative number.
  - Terminate computation $\Rightarrow$ unbounded solution.
- **Step 6:** Update the basic solution

  $\overline{x}_i = x_i - \theta \underline{y}_i$; $i \neq l$; Set $x_l = \theta$ corresponding to the new basic variable, k (l goes out)
- **Step 7:** Update the basis and return to Step 1.

Copyright ©2004 by K. Pattipati

Phase I of LP

- How to get initial feasible solution…Phase I of LP
  - An LP problem for Phase I
    - $\min\left(\sum_{i=1}^{m}\hat{y}_i\right)$ such that $A\underline{x}+I_m\,\underline{\hat{y}}=\underline{b};\quad \underline{x}\geq\underline{0},\ \ \underline{\hat{y}}\geq 0$

      $\underline{\hat{y}}$ ~ Artificial Variable
    - If we can find an optimal solution such that $\sum_{i=1}^{m}\hat{y}_i=0$, then we have $\underline{x}_B$.
    - If $\sum_{i=1}^{m}\hat{y}_i>0$ then there is no feasible solution to $A\underline{x}=\underline{b},\ \underline{x}\geq\underline{0}$.

      $\Rightarrow$ Infeasible Problem
    - Solve via revised simplex starting with $\underline{x}=\underline{0},\ \underline{\hat{y}}=\underline{b}$ & $B=I_m$.
  - **Another approach is to combine both phases I and II by solving:**
    - $\min_{\underline{x},\underline{y}}\left(e^T\underline{x}+Me^T\underline{y}\right)$ (where M is a large number)

      s.t. $A\underline{x}+\underline{y}=\underline{b};\quad \underline{x}\geq 0,\quad \underline{y}\geq 0$
    - **This is called the "big-M" method.**

Copyright ©2004 by K. Pattipati

Basis Updates

How to Update Basis:

- **NOTE: We need to solve:**
  - $B^T \underline{\lambda} = \underline{c}_B$ and $B\underline{y} = \underline{a}_k$, where the B's differ by only one column between any two subsequent iterations $\Rightarrow$ column $\underline{a}_k$ replaces $\underline{a}_l$
- A simple way to solve these equations is to propagate B^{-1} from one iteration to the next.
  - Recall: $B_{new} = B_{old} - \text{column } \underline{a}_l + \text{column } \underline{a}_k = B_{old} + (\underline{a}_k - \underline{a}_l)\underline{e}_l^T \quad \Rightarrow$ rank one update
  - So, $B_{new}^{-1} = B_{old}^{-1} - \dfrac{B_{old}^{-1}(\underline{a}_k - \underline{a}_l)\underline{e}_l^T B_{old}^{-1}}{1 + \underline{e}_l^T B_{old}^{-1}(\underline{a}_k - \underline{a}_l)} = \left[I - \dfrac{B_{old}^{-1}(\underline{a}_k - \underline{a}_l)\underline{e}_l^T}{y_l}\right] B_{old}^{-1}$. NOTE: $B_{old}^{-1}\underline{a}_k = \underline{y}$ and $B_{old}^{-1}\underline{a}_l = \underline{e}_l$

$\Rightarrow B_{old}^{-1} = EB_{new}^{-1} = $ product form of the inverse (PFI)

$$\text{where } E = I - \frac{\underline{y}\underline{e}_l^T}{y_l} + \frac{1}{y_l}\underline{e}_l\underline{e}_l^T = \begin{bmatrix} 1 & 0 & \dots & -y_1/y_l & \dots & 0 \\ 0 & 1 & \dots & -y_2/y_l & \dots & 0 \\ 0 & \dots & \dots & 1/y_l & \dots & 0 \\ 0 & \dots & \dots & -y_m/y_l & \dots & 1 \end{bmatrix}$$

***E* is called an "Elementary Matrix."**

  - For large scale problems, store E as a vector and update $\underline{\lambda}^T$ and $\underline{p}^T$ sequentially as follows:

$$\underline{\lambda}^T = \left[\left(\underline{c}_B^T E_p\right)E_{p-1}\right]\dots E_1 \quad \text{or} \quad \underline{y} = E_p\left[\dots \dots \left(E_2\left(E_1\underline{a}_k\right)\right)\dots\right]$$

- **What if y_l is small? This creates a problem…**
- **Modern revised simplex methods use LU or QR decompositions.**

Copyright ©2004 by K. Pattipati

Sequential LU and QR

❑ LU Decomposition

- **If B is the current basis:**
  - $B = \left(\underline{a}_1\ \underline{a}_2\ \dots\ \underline{a}_m\right) = L_{old} U_{old} = B_{old}$
  - $B_{new} = \left(\underline{a}_1\ \underline{a}_2\ \dots\ \underline{a}_{l-1}\ \underline{a}_{l+1} \cdot \underline{a}_m\ \underline{a}_k\right)$
  - NOTE: $H = L_{old}^{-1} B_{new} = \left[\underline{u}_1 \underline{u}_2 \dots \underline{u}_{l-1} \underline{u}_{l+1} \dots \underline{u}_m L_{old}^{-1} \underline{a}_k\right]$ is an upper Hessenberg matrix.
- **Use a sequence of elimination steps on H to get:**
  - $U_{new} = \hat{M}_{m-1} \dots \hat{M}_{l+1} \hat{M}_l H \quad \Rightarrow B_{new} = L_{old} \hat{M}_l^{-1} \dots \hat{M}_{m-1} U_{new}$
  - Store: $L_{new}^{-1} = \hat{M}_{m-1} \dots \hat{M}_l L_{old}^{-1}$

❑ QR Decomposition

- $B_{new} = \left(\underline{a}_1 \underline{a}_2\ \dots\ \underline{a}_{l-1} \underline{a}_{l+1} \cdot \underline{a}_m \underline{a}_k\right); \qquad Q_{old}^T B_{new} = H$
- Do Givens on H :

  $J_{m-1}^T \dots J_l^T H = R_{new}$ and $Q_{new} = Q_{old} J_l \dots J_{m-1}$
- Theoretically, revised simplex is an exponential algorithm $O\left(\binom{n}{m}\right)$.
- In practice, it takes approximately $2(n+m)$ iterations.
- Each iteration takes approximately $O\left(m^2 + m(n-m)\right)$ operations.

Copyright ©2004 by K. Pattipati

Sensitivity Analysis

Duality and Sensitivity Analysis

- Recall that the basic feasible solution $\underline{x} = \begin{bmatrix} \underline{x}_B \\ \underline{x}_N \end{bmatrix} = \begin{bmatrix} B^{-1}\underline{b} \\ \underline{0} \end{bmatrix}$

  is the solution of SLP "min $\underline{c}^T \underline{x}$ s.t. $A\underline{x} = \underline{b}$, $\underline{x} \geq \underline{0}$" if and only if:

  – $\underline{\lambda}^T = \underline{c}_B^T B^{-1}$ ~ Vector of simplex (Lagrange) multipliers or dual variables

  – $\underline{p}^T = \underline{c}_N^T - \underline{c}_B^T B^{-1} N \geq \underline{0}$ ~ Non-negative relative cost vector

- **Note that the optimal cost is given by**

$$z = \underline{c}^T \underline{x} = \underline{c}_B^T B^{-1} \underline{b} = \underline{\lambda}^T \underline{b}$$

  – So, z can be gotten by knowing optimal $\underline{x}_B$ or optimal $\underline{\lambda}$.

  – Q: Is there another way to get $\underline{\lambda}$?

  – A: Yes, by solving an equivalent LP, called a dual LP problem.

Copyright ©2004 by K. Pattipati

Dual LP Problems

Dual of an SLP

Primal Problem	Dual Problem
$\min\limits_{\underline{x}} \left(\underline{c}^T \underline{x}\right)$	$\max \underline{\lambda}^T \underline{b}$
s.t. $A\underline{x} = \underline{b},\ \underline{x} \geq \underline{0}$	s.t. $A^T \underline{\lambda} \leq \underline{c}$
n variables	n constraints
m constraints	m variables
non-negativity constraints on $\underline{x}$	no constraints on the sign of $\{\lambda_i\}$

ASYMMETRIC DUAL

- **Duality of an Inequality constrained LP (InLP)**

Primal	Dual
$\min\limits_{\underline{x}} \underline{c}^T \underline{x}$	$\min\limits_{\underline{\lambda}} \underline{\lambda}^T \underline{b}$
s.t. $A\underline{x} \geq \underline{b}$	s.t. $A^T \underline{\lambda} \leq \underline{c}$
$\underline{x} \geq \underline{0}$	$\underline{\lambda} \geq \underline{0}$

SYMMETRIC DUAL

Copyright ©2004 by K. Pattipati

Duality Properties

- **Dual of a Dual = Primal**
  - For any feasible $\underline{x}$ and dual feasible $\underline{\lambda}$

$$\left.\begin{aligned}&\text{(SLP): } \underline{\lambda}^T \underline{b} = \underline{\lambda}^T A\underline{x} \le \underline{c}^T \underline{x}\\ &\text{(InLP): } \underline{\lambda}^T \underline{b} \le \underline{\lambda}^T A\underline{x} \le \underline{c}^T \underline{x}\end{aligned}\right\} \text{ weak duality lemma}$$

$$\boxed{\text{Dual feasible solution } \le \text{primal feasible solution}}$$

- Very useful concept in deriving efficient algorithms for large integer programming problems (e.g., scheduling) with separable structures.

Complementary Slackness Conditions

1) $\left(\underline{c}^T - \underline{\lambda}^T A\right)\underline{x} = 0 \;\Rightarrow\; p_j = c_j - \underline{\lambda}^T \underline{a}_j = 0 \text{ or } x_j = 0$

2) $\underline{\lambda}^T\left(A\underline{x} - \underline{b}\right) = 0 \;\Rightarrow\; q_i = \underline{a}_i^T \underline{x} - b_i = 0 \text{ or } \lambda_i = 0$

   – $\underline{\lambda}^T \underline{a}_j$ ~ synthetic cost of variable j

- **For variables in the optimal basis, relative cost $p_j = 0 \Rightarrow$ synthetic cost = real cost**
- **For variables not in optimal basis, relative cost $p_j \ge 0 \Rightarrow$ synthetic cost $\le$ real cost**

Copyright ©2004 by K. Pattipati

Simplex Multipliers & Sensitivity - 1

Interpretation of Simplex Multipliers

- Suppose $\underline{b} \rightarrow \underline{b} + \delta\underline{b}$ without changing the optimal basis.
- Change in the optimal objective function value

$$\delta z = \underline{c}_B^T B^{-1} \delta \underline{b} = \underline{\lambda}^T \delta \underline{b}$$

- $\lambda_i = \dfrac{\delta z}{\delta b_i} =$ marginal price (value) of the i^{th} resource (i.e., right hand side of b_i)
- $\{\lambda_i\}$ are also called shadow prices, dual variables,

  Lagrange multipliers, or equilibium prices.

- Sensitivity (post-optimality) analysis
  - Q: How much can we change $\{c_i\}$ & $\{b_i\}$ without changing the optimal basis?
  - Consider:

$$\min_{\underline{x}} \left(\underline{c} + \alpha \underline{d}\right)^T \underline{x}; \quad \text{s.t. } A\underline{x} = \underline{b}, \quad \underline{x} \geq \underline{0}$$

  - α is the parameter to be varied
  - Nominal value of $\alpha = 0$.
  - $\underline{d} = \underline{e}_j \quad \Rightarrow$ Want to find the range for the j^{th} coefficient.

Copyright ©2004 by K. Pattipati

Simplex Multipliers & Sensitivity - 2

- **Fact:** Basis B will be optimal as long as nonbasic reduced costs $\{p_k\}$ remain non-negative (recall that the reduced costs for basic variables are zero).
  - Split $\underline{c}$ and $\underline{d}$ as $\underline{c}^T = \left(\underline{c}_B^T \mid \underline{c}_N^T\right)$ and $\underline{d}^T = \left(\underline{d}_B^T \mid \underline{d}_N^T\right)$
  - The required condition is:

$$\left(\underline{c}_N^T - \underline{c}_B^T B^{-1} N\right) + \left(\underline{d}_N^T - \underline{d}_B^T B^{-1} N\right) \geq 0$$

$$\underline{p}^T + \underline{q}^T \geq 0 \quad \Rightarrow \quad \underline{q}^T \geq -\underline{p}^T$$

  - So the range of $\alpha = \left(\alpha_{\min}, \alpha_{\max}\right)$, where

$$\alpha_{\min} = \max\left\{ \max\left\{ \frac{-p_j}{q_j} :\ q_j > 0 \text{ and } j \text{ is nonbasic} \right\}, -\infty \right\}$$

$$\alpha_{\max} = \min\left\{ \min\left\{ \frac{-p_j}{q_j} :\ q_j < 0 \text{ and } j \text{ is nonbasic} \right\}, \infty \right\}$$

- If $\alpha \in (\alpha_{min}, \alpha_{max})$, the new optimal cost is:

$$z(\alpha) = (\underline{c}_B{}^T + \alpha \underline{d}_B{}^T).B^{-1}\underline{b} = z(0) + \alpha \underline{d}_B{}^T \underline{x}_B$$

Copyright ©2004 by K. Pattipati

Simplex Multipliers & Sensitivity - 3

- Consider parametric changes in $\underline{b}$

$$\begin{aligned}&\min_{\underline{x}} \; \underline{c}^T \underline{x}\\ &\text{s.t. } A\underline{x} = \underline{b} + \alpha \underline{d} \quad ; \quad \alpha = 0 \;\; \text{nominal}\\ &\qquad \underline{x} \geq \underline{0}\end{aligned}$$

- If B is the optimal basis, then need

$$\underline{x}^T = (\underline{x}_B^T \;\; \underline{x}_N^T) = [\underline{\bar{b}}^T + \alpha \underline{\bar{d}}^T, 0]$$

$$\text{where } \underline{\bar{b}}^T = B^{-1}\underline{b} \text{ and } \underline{\bar{d}}^T = B^{-1}\underline{d}$$

- The range of $\alpha = (\alpha_{min}, \alpha_{max})$ is given by:

$$\alpha_{\min} = \max\left\{\max_{1\le i\le m}\{\frac{-\bar{b}_i}{\bar{d}_i} : \bar{d}_i > 0\}, -\infty\right\}; \; \alpha_{\max} = \min\left\{\min_{1\le i\le m}\{\frac{-\bar{b}_i}{\bar{d}_i} : \bar{d}_i < 0\}, \infty\right\}$$

- If $\alpha \in (\alpha_{min}, \alpha_{max})$, then

$$z(\alpha) = \underline{c}_B^T B^{-1}(\underline{b} + \alpha \underline{d}) = \underline{\lambda}^T(\underline{b} + \alpha \underline{d}) = z(0) + \alpha \underline{\lambda}^T \underline{d}$$

Copyright ©2004 by K. Pattipati

Karmarkar's Interior Point Method

- Karmarkar's Interior Point Algorithm
  - Discuss not the original Karmarkar's algorithm, but an equivalent (and more general) formulation based on **barrier functions**

$$\begin{array}{llll} & \min_{\underline{x}} \underline{c}^T \underline{x} & & \min_{\underline{x}} f(\underline{x},\mu) = \underline{c}^T \underline{x} - \mu \sum_{j=1}^{n} \ln x_j;\ \mu > 0 \\ \text{SLP:} & \text{s.t.}\quad A\underline{x} = \underline{b} & \Rightarrow \text{Barrier} & \text{s.t.}\quad A\underline{x} = \underline{b} \\ & \quad\quad \underline{x} \geq \underline{0} & \quad \text{Problem} & \\ & \text{optimal solution } \underline{x}^* & & \text{optimal solution } \underline{x}^*(\mu) \end{array}$$

  - Key: $\underline{x}^*(\mu) \rightarrow \underline{x}^*$ as the barrier parameter $\mu \rightarrow 0$
  - $\exists$ many variations of barrier function formulations. Wewill discuss them later

Copyright ©2004 by K. Pattipati

Newton's Method for NLP

- Consider the general NLP

$$\min_{\underline{x}} f(\underline{x}) \text{ s.t. } A\underline{x} = \underline{b}$$

  - Suppose $\underline{x}$ is feasible, then $\bar{\underline{x}} = \underline{x} + \alpha \underline{d}$

    $\underline{d} \sim$ search direction

  - Pick $\alpha \ni A\bar{\underline{x}} = \underline{b}$ (new point is feasible) and $f(\underline{x}) < f(\bar{\underline{x}})$

- What does Newton's Method do for this problem?

  - Feasibility $\Rightarrow A\bar{x} = A\underline{x} + \alpha A\underline{d} = 0 \Rightarrow A\underline{d} = 0$
  - Newton's method fits a quadratic to $f(\underline{x})$ at the current point and takes $\alpha = 1$
  - $f(\underline{x} + \underline{d}) = f(\underline{x}) + \underline{g}^T \underline{d} + 1/2\, \underline{d}^T H \underline{d}$ where $\underline{g} = \nabla \underline{f}(\underline{x})$; $H = \nabla^2 f(\underline{x})$
  - Newton's method solves a quadratic problem to find $\underline{d}$

  ($\Rightarrow$ a weighted least squares problem)

Copyright ©2004 by K. Pattipati

Optimality Conditions

- Consider

$$\min_{\underline{d}} \underline{g}^T \underline{d} + 1/2\, \underline{d}^T H \underline{d} \;\Rightarrow\; \min_{\underline{d}} 1/2 \,\| H^{1/2} d - H^{1/2} \underline{g} \|_2^2;\; H^{1/2} \text{ symmetric squareroot}$$
$$\text{s.t.} \quad A\underline{d} = \underline{0} \qquad\qquad \text{s.t.} \quad A\underline{d} = \underline{0}$$

- Define Lagragian function:

$$L(\underline{d}, \underline{\lambda}) = \underline{g}^T \underline{d} + 1/2\, \underline{d}^T H \underline{d} - \underline{\lambda}^T \underline{d} \;;\; \underline{\lambda} \sim \text{Lagrange multiplier}$$

- Karush-Kuhn-Tucker necessary conditions of optimality:

$$\Rightarrow \partial L / \partial \underline{d} = 0 \;\Rightarrow\; \underline{g} + H\underline{d} - A^T \underline{\lambda} = \underline{0}$$
$$\partial L / \partial \underline{\lambda} = 0 \;\Rightarrow\; -A\underline{d} = 0;\; \lambda = (AH^{-1}A^T)^{-1} AH^{-1} \underline{g}$$

- **Special NLP = barrier formulation of LP:**

$$\underline{g} = \nabla f(\underline{x}) = \underline{c} - \mu D^{-1} \underline{e} \text{ and } H = \nabla^2 f(\underline{x}) = \mu D^{-2}$$

where $D = Diag(x_j)\;, j \;=\; 1,\; 2,\; \ldots,\; n$ and $\underline{e} = (1\,1\,1\,\ldots\,1)^T$

Copyright ©2004 by K. Pattipati

Optimality Conditions for LP

- Karush-Kuhn-Tucker conditions for special NLP are:

$$\mu D^{-2}\underline{d} + (\underline{c} - \mu D^{-1}\underline{e} - A^T\underline{\lambda}) = \underline{0}$$
$$A\underline{d} = \underline{0}$$

- So,

$$\underline{d} = \frac{-1}{\mu} D^2(\underline{c} - \mu D^{-1}\underline{e} - A^T\underline{\lambda}) \qquad (1)$$

- Using $A\underline{d} = \underline{0}$ in (1), we get

$$\underline{\lambda} = (AD^2A^T)^{-1}AD^2(\underline{c} - \mu D^{-1}\underline{e}) \qquad (2)$$

- So, λ is the solution of weighted least square (WLS) problem:

$$\min_{\underline{\lambda}} \| D[\underline{c} - \mu D^{-1}\underline{e} - A^T\underline{\lambda}] \|_2^2$$

Copyright ©2004 by K. Pattipati

Barrier Function Algorithm

Barrier Function Algorithm

Choose a strictly feasible solution and constant $\mu > 0$. Let the tolerance parameter be ε and a parameter associated with the update of μ be σ.

For $k = 0, 1, 2, \ \ldots$ DO

Let $D = Diag(x_j)$

Compute the solution $\underline{\lambda}$ to

$$(AD^2A^T)\underline{\lambda} = AD^2(\underline{c} - \mu D^{-1}\underline{e}) \ \ldots \text{ WLS solution}$$

Let

$$\underline{p} = \underline{c} - A^T\underline{\lambda}$$

$$\underline{d} = -D^2(\underline{p} - \mu D^{-1}\underline{e}) / \mu$$

$$\underline{x} = \underline{x} + \underline{d}$$

If $\underline{x}^T\underline{p} < \varepsilon$, stop: $\underline{x}$ is near-optimal solution … complementary slackness condition.

else

$$\mu = \left(1 - \frac{\sigma}{\sqrt{n}}\right)\mu$$

end if

end DO

Copyright ©2004 by K. Pattipati

Practicalities & Insights -1

1) Finding a feasible point

- Select any $\underline{x}_0 > \underline{0}$ and define $\xi_0 \underline{s} = \underline{b} - A\underline{x}_0$ with $\|\underline{s}\|_2 = 1$
  $\Rightarrow \xi_0 = \|b - A\underline{x}_0\|_2$ and solve

$$\min_{\underline{x},\xi} \ \xi \quad s.t. \ (A \ \underline{s})\begin{pmatrix} \underline{x} \\ \xi \end{pmatrix} = \underline{b}; \ x \geq 0, \ \xi \geq 0$$

- Thesolution : $\xi = 0$ or when ξ starts becoming negative stop
- Suggest $\underline{x}_0 = \|b\|\underline{e}$

2) Since the method uses Newton's directions, expect quadratic convergence near minimum

3) Major computational step: Least-squares subproblem

$$AD^2A^T\underline{\lambda} = AD^2(\underline{c} - \mu D^{-1}\underline{e})$$

Generally A is sparse

We will discuss the computational aspects of Least-squares subproblem later

Copyright ©2004 by K. Pattipati

Practicalities & Insights - 2

4) The algorithm (theoretically) requires $O(\sqrt{n}L)$ iterations with overall complexity where $O(n^3 L)$

$$L = \sum_{i=0}^{m} \sum_{j=1}^{n} \left[\log |a_{ij}| + 1 \right] + 1$$

5) In practice, the method typically takes 20-50 iterations even for very large problems (>20,000 variables). Simplex, on the other hand, takes increasingly large number of iterations with the problem size, n.

6) Initialize $\mu = 2^{O(L)}$ and $\sigma \cong 1/6$. In practice, need to experiment with the parameters.

7) Other potential functions : $f(\underline{x}, q) = r \ln(\underline{c}^T \underline{x} - q) - \sum_j \ln x_j$

where

$r = n + \sqrt{n}$ and

$q =$ a lower -bound on the optimal cost

Copyright ©2004 by K. Pattipati

Practicalities & Insights - 3

- Variants of the algorithm
  - Problem with barrier function approach:
    - Update of μ
    - Selection of initial μ and parameter σ
  - $\exists$ two classes of algorithms
    - Affine scaling
    - Power series approximation
      - Views affine scaling directions as a set of differential equations
      - Not competitive with affine scaling methods
  - Do not know if the variants have polynomial complexity. But, they work well in practice!!

Copyright ©2004 by K. Pattipati

Affine Scaling Method - 1

- Affine scaling:
  - Typically, the affine scaling methods are used on the dual problem

Primal	Dual	Modified dual
$\min\limits_{\underline{x}} \; \underline{c}^T \underline{x}$	$\max\limits_{\underline{\lambda}} \; \underline{\lambda}^T \underline{b}$	$\max\limits_{\underline{\lambda}} \; \underline{\lambda}^T \underline{b}$
s.t. $A\underline{x} = \underline{b}$ $\Leftrightarrow$	s.t. $A^T \underline{\lambda} \le \underline{c}$ $\Leftrightarrow$	s.t. $A^T \underline{\lambda} + \underline{p} = \underline{c}$
$\underline{x} \ge \underline{0}$		$\underline{p} \ge \underline{0}$

  - Suppose have a strictly feasible $\underline{\tilde{\lambda}}$ and the corresponding reduced cost vector (slack vector) $\underline{\tilde{p}}$
  - Define

$$\underline{\hat{p}} = P^{-1} \underline{p}, \text{ where } \quad P = Diag(\tilde{p}_1, \tilde{p}_2 \; , \; \ldots \; , \tilde{p}_n)$$

  - So, the dual problem is :

$$\max \; \underline{\lambda}^T \underline{b} \quad \text{s.t.} \;\; A^T \underline{\lambda} + P\underline{\hat{p}} = c; \;\; \underline{\hat{p}} \ge \underline{0}$$

Copyright ©2004 by K. Pattipati

Affine Scaling Method - 2

- From the equality constraint:

$$\underline{\hat{p}} = P^{-1}(c - A^T \underline{\lambda}) \;\Rightarrow\; P^{-1}A^T \underline{\lambda} = (P^{-1}\underline{c} - \underline{\hat{p}})$$

- Assuming full column rank of A^T or row rank of A

$\Rightarrow$ linearly independent constraints in primal

$$AP^{-2}A^T \underline{\lambda} = AP^{-1}(P^{-1}\underline{c} - \underline{\hat{p}})$$

$$\Rightarrow \underline{\lambda} = (AP^{-2}A^T)^{-1} AP^{-1}(P^{-1}\underline{c} - \underline{\hat{p}}) = M(P^{-1}\underline{c} - \underline{\hat{p}})$$

- Note that $\underline{\lambda} \in R(AP^{-1}) = R(M)$
- Eliminating $\underline{\lambda}$ from the dual problem, we have:

$$\begin{array}{llcll} \max_{\underline{\hat{p}}} & \underline{b}^T M(P^{-1}\underline{c} - \underline{\hat{p}}) = f(\underline{\hat{p}}) & & \min_{\underline{\hat{p}}} & \underline{b}^T M \underline{\alpha} \\ \text{s.t.} & H(\underline{\hat{p}} - P^{-1}\underline{c}) = \underline{0} & \Leftrightarrow & \text{s.t.} & H\underline{\alpha} = \underline{0} \\ & \underline{\hat{p}} \geq \underline{0} & & \text{where} & \underline{\alpha} = \underline{\hat{p}} - P^{-1}\underline{c} \end{array}$$

and where

$H = I - P^{-1}A^T M$, asymmetric projection matrix

$$\Rightarrow H^2 = H$$

Copyright ©2004 by K. Pattipati

Affine Scaling Method - 3

- In addition, we have

$$AP^{-1}H = 0 \;\Rightarrow\; \text{columns of } H \in N(AP^{-1})$$

- Note that we want $\underline{\alpha} \in N(H) \Rightarrow \alpha \in R(P^{-1}A^T)$
- But, $R(P^{-1}A^T) = R(M^T)$
- The gradient of $f(\underline{\widehat{p}})$ w.r.t. scaled reduced costs $\underline{\widehat{p}}$ is

$$\underline{\hat{g}}_p = -M^T \underline{b} \in R(M^T) = R(P^{-1}A^T)$$

$\Rightarrow$ Result: The gradient w.r.t. scaled reduced costs, $\underline{\widehat{p}}$, already lies in the range space of $P^{-1}A^T$, making projection unnecessary.

- In terms of original unscaled reduced costs, the project gradient is

$$\boxed{\underline{g}_p = -P\underline{\widehat{g}}_p = -A^T(AP^{-2}A^T)^{-1}\underline{b}}$$

Copyright ©2004 by K. Pattipati

Affine Scaling Method - 4

- The corresponding feasible direction with respect to $\underline{\lambda}$ is:

$$\boxed{\underline{d}_{\lambda} = -MM^{T}\underline{\hat{g}}_{p} = (AP^{-2}A^{T})^{-1}\underline{b}}$$

$$\Rightarrow \boxed{\underline{g}_{p} = -A^{T}\underline{d}_{\lambda}}$$

- If $\underline{g}_{p} \geq \underline{0} \Rightarrow$ dual problem is unbounded $\Rightarrow$ primal is infeasible (assuming $\underline{b} \neq \underline{0}$)

- Otherwise, we replace λ by

$$\lambda \leftarrow \underline{\lambda} + \alpha\underline{d}_{\lambda}$$

where

$$\alpha = \beta\alpha_{\max}; \beta \approx 0.95$$

$$\alpha_{\max} = \min\left\{\frac{-p_i}{g_{pi}} : g_{pi} < 0, i = 1, 2, \ldots, n\right\}$$

- Note that primal solution $\underline{x}$ is:

$$\underline{x} = -P^{-2}g_{p} = -P^{-2}A^{T}(AP^{-2}A^{T})^{-1}\underline{b}$$

since it satisfies $A\underline{x} = \underline{b}$.

Copyright ©2004 by K. Pattipati

Dual Affine Scaling Algorithm

- Dual affine scaling algorithm

Start with a strictly feasible $\underline{\lambda}$, stopping criterion ε and β.

$z_{old} = \underline{\lambda}^T \underline{b}$

For $k = 0, 1, 2, \ldots$ DO

$$\underline{p} = \underline{c} - A^T \underline{\lambda}; \quad P = Diag(p_1\ p_2\ \ldots\ p_n)$$

Compute the solution $\underline{d}_\lambda$

$(AP^{-2}A^T)\underline{d}_\lambda = \underline{b}; \quad \underline{g}_p = -A^T \underline{d}_\lambda$

If $\underline{g}_p \geq 0$

Stop: unbounded dual solution $\Rightarrow$ primal is infeasible

else

$$\alpha = \beta \min\left\{\frac{-p_i}{g_{pi}} : g_{pi} < 0,\ i = 1, 2, \ldots, n\right\}$$

$\underline{\lambda} \leftarrow \underline{\lambda} + \alpha \underline{d}_\lambda (\Rightarrow \underline{p} \leftarrow \underline{p} + \alpha \underline{g}_p$ next step); $z_{new} = \underline{\lambda}^T \underline{b}$

If $\dfrac{|z_{new} - z_{old}|}{\max(1, |z_{old}|)} < \varepsilon$

stop: found an optimal solution $\underline{x} = -P^{-2} \underline{g}_p$

else

$z_{old} \leftarrow z_{new}$

end if

end if

end DO

Copyright ©2004 by K. Pattipati

Initial Feasible Solution

- Finding an initial strictly feasible solution for the dual affine scaling algorithm

$$\underline{\lambda}_0 = \left(\frac{\| \underline{c} \|_2}{\| A^T \underline{b} \|_2} \right) \underline{b}$$

- Want to find a $\underline{p} \ni \underline{p} = -\xi \underline{e}$
- Select initial ξ_0 as

$$\xi_0 = -2 \min \left\{ \left(\underline{c} - A^T \underline{\lambda} \right)_i : i = 1, 2, \ldots, m \right\}$$

- Solve an $(m+1)$ variable LP:

$$\max_{\underline{\lambda},\, \xi} \quad \underline{\lambda}^T \underline{b} - \mu \xi \quad \text{s.t.} \quad A^T \underline{\lambda} - \xi \underline{e} < \underline{c}$$

- Select $\mu = \gamma . \dfrac{\underline{\lambda}_0^T \underline{b}}{\xi_0}; \quad \gamma = 10^5$
- The initial $(\underline{\lambda}_0, \xi_0)$ are feasible for the problem
- Notes:
  - If $\xi < 0$ at iteration $k \Rightarrow$ found a feasible $\underline{\lambda}$
  - If the algorithm is such that optimal $\xi < \varepsilon \Rightarrow$ dual is infeasible $\Rightarrow$ primal is unbounded

Copyright ©2004 by K. Pattipati

Least Squares Subproblem

- Lease-squares subproblem: implementation issues
  - Generally A is sparse
  - Major computational step at each iteration

$$AP^{-2}A^T\underline{d} = \underline{b} \quad \text{... Affine scaling}$$

$$AD^2A^T\underline{\lambda} = AD^2(\underline{c} - \mu D^{-1}\underline{e}) = AD(D\underline{c} - \mu\underline{e})$$

... barrier function method

  - Key: need to solve a symmetric positive definite system $\Sigma\underline{y} = \underline{b}$

Copyright ©2004 by K. Pattipati

Solution Approaches - 1

- Solution approaches:
- Direct methods:

a) Cholesky factorization: $\Sigma = SS^T,\ S = \text{lower } \Delta$

b) LDL^T factorization: $\Sigma == LDL^T,\ S = \text{unit lower } \Delta$

c) QR factorization: of $P^{-1}A^T$ or DA^T

- Methods to speed up factorization
  - During each iteration only D or P^{-1} changes, while A remains unaltered
    - Nonzero structure of Σ is static throughout.
    - So, during the first iteration, keep track of the list of numerical operations performed
  - Perform factorization only if the diagonal scaling matrix has changed significantly
  - Consider
    - $\Sigma == AP^{-2}A^T$
    - replace P by $\overline{P}$

Copyright ©2004 by K. Pattipati

Solution Approaches - 2

where

$$\bar{P}_{ii}^{new} = \begin{pmatrix} \bar{P}_{ii}^{old} & \text{if } |P_{ii} - \bar{P}_{ii}^{old}| / |\bar{P}_{ii}^{old}| < \delta \\ P_{ii} & \text{otherwise} \end{pmatrix}$$

$$\delta \sim 0.1$$

$$\text{define } \Delta P_{ii} = \bar{P}_{ii}^{new} - \bar{P}_{ii}^{old}$$

$$\text{then } \Sigma^{new} = \Sigma^{old} + \sum_{i:\Delta P_{ii} \neq 0} \Delta P_{ii} . \underline{a}_i . \underline{a}_i^T$$

$$\underline{a}_i = i^{th} \text{ column of } A$$

- – So, use rank-one modification methods discussed in Lecture 8

- Perform pivoting to reduce fill-ins $\Rightarrow$ having nonzero elements in factors where there are zero elements in Σ .
  - – Recall that $P \Sigma P^T P \underline{y} = P \underline{b}$
  - – Unfortunately, finding the optimal permutation matrix to reduce filled-in is NP-complete
  - – However, $\exists$ heuristics
    - ▪ minimum degree
    - ▪ minimum local fill-in

Copyright ©2004 by K. Pattipati

Solution Approaches - 3

- Combine with an iterative method if have a few dense columns in A that will make impracticably dense Σ. (Recall the outer product representation)

  $\Rightarrow$ Hybrid factorization and conjugate gradient method called a preconditioned conjugate gradient method.

  Idea: At iteration k, split columns of A into two parts S and $\bar{S}$

  where columns of A_s are sparse (i.e., have density < $\lambda(\approx 0.3)$)

  – Form $A_s P^{-2} A_s^T$

  – Find <u>incomplete</u> Cholesky factor L such that

  $$Z_s = A_s P^{-2} A_s^T = LL^T$$

  – Basically the idea is to step through the Cholesky decomposition, but setting $l_{ij} = 0$ if the corresponding $\Sigma_{sij} = 0$

Copyright ©2004 by K. Pattipati

Incomplete Cholesky Algorithm - 1

- Incomplete Cholesky Algorithm

For $k = 1, \ldots, m$ DO

$l_{kk} = \sqrt{\Sigma_{skk}}$

For $i = k+1, \ldots, m\ DO$

If $\Sigma_{sik} \neq 0$

$l_{ik} = \Sigma_{sik} / l_{kk}$

end if

end DO

For $j = k+1, \ldots, m$ DO

For $i = j, \ldots, m$ DO

If $\Sigma_{sij} \neq 0$

$\Sigma_{sij} = \Sigma_{sij} - l_{ik} l_{jk}$

end if

end DO

end DO

end DO

Copyright ©2004 by K. Pattipati

Incomplete Cholesky Algorithm - 2

- Now consider the original problem

$$\Sigma \underline{y} = A^T P^{-2} A \underline{y} = \underline{b}$$

$$L^{-1}\Sigma(L^{-1})^T . L^T \underline{y} = L^{-1}\underline{b}$$

$$\Rightarrow \quad Q\underline{u} = \underline{f}$$

where

$$Q = L^{-1}\Sigma(L^{-1})^T; \; \underline{u} = L^T \underline{y}; \; \underline{f} = L^{-1}\underline{b}$$

- Solve $Q\underline{u} = \underline{f}$ via conjugate gradient algorithm (see Lecture 5)

Copyright ©2004 by K. Pattipati

Conjugate Gradient Algorithm

- Conjugate gradient algorithm

$\underline{u} = \underline{f}$ ... initial solution

$c = \| f \| 2$ ... norm of RHS

$\underline{r} = f - Q\underline{u}$ ... initial residual (negative gradient of $(\frac{1}{2} u^T Q u - u^T f)$)

$\rho = \| r \|$ ... square of norm of initial residual

$\underline{d} = \underline{r}$ ... initial direction

$k = 0$

do while $\sqrt{\rho} / c \geq \varepsilon$ and $k \leq k_{\max}$

$\underline{w} = Q\underline{d}$

$\alpha = r / \underline{d}^T Q \underline{d}$ ... step length

$u = u + \alpha \underline{d}$ ... new solution

$\underline{r} = \underline{r} - \alpha \underline{w}$ ... new residual, $\underline{r} = f - Q\underline{u}$

$\beta = \| r \|_2^2 / \rho$ ... parameter to update direction

$\underline{d} = r + \beta \underline{d}$ ... new direction

$\rho = \| r \|_2^2$

$k = k + 1$

end DO

- Computational load: $O(m^2 + 10m)$
- Need to store only for vector: $\underline{u}$, $\underline{r}$, $\underline{d}$ and $\underline{w}$

Copyright ©2004 by K. Pattipati

Simplex vs. Dual Affine Scaling - 1

- ❑ Comparison of simplex and dual affine scaling methods
  - Three types of test problems
- ❑ NETLIB test problems
  - 31 test problems
  - The library and test problem can be accessed via electronic mail

    netlib@anl-mcs (ARPANET/CSNET)

    (or) research ! netlib (UNIX network)
  - # of variables n ranged from 51 to 5533
  - # of constraints m ranged from 27 to 1151
  - # of non-zero elements in A ranged from 102 to 16276
  - Comparisons on IBM 3090

Copyright ©2004 by K. Pattipati

Simplex vs. Dual Affine Scaling - 2

	Simplex	Affine scaling
Iterations	(6, 7157)	(19,55)
Ratio of time per iteration	(0.093, 0.356)	1
Total CPU time range (secs)	(0.01, 217.67)	(0.05, 31.70)
Ratio of CPU times (simplex/Affine)	(0.2, 10.7)	1

- Multi-commodity Network Flow problems
  - Specialized LP algorithms exist that are better than simplex
  - ∃ a program to generate random multi-commodity network flow problem called *MNETGN*
  - 11 problems were generated
  - # of variables n in the range (2606, 8800)
  - # of constraints m in the range (1406, 4135)
  - Non-zero elements in A ranged from 5212 to 22140

Copyright ©2004 by K. Pattipati

Simplex vs. Dual Affine Scaling - 3

	Simplex MINOS 4.0	Specialized Simplex (MCNF 85)	Affine scaling
Total # of iterations	(940, 21915)	(931, 16624)	(28, 35)
Ratios of time per iteration (w.r.t. Affine scaling)	(0.010, 0.069)	(0.0018, 0.0404)	1
Total cpu time (secs)	(12.73, 1885.34)	(7.42, 260.44)	(6.51, 309.50)
Ratios of cpu times w.r.t. affine scaling	(1.96, 11.56)	(0.59, 4.15)	1

Copyright ©2004 by K. Pattipati

Simplex vs. Dual Affine Scaling - 4

- Timber Harvest Scheduling problems
  - 11 timber harvest scheduling problems using a program called FORPLAN
  - # of variables ranged from 744 to 19991
  - # of constraints ranged from 55 to 316
  - # of nonzero elements in A ranged from 6021 to 176346

	Simplex (MINOS 4.0) (default pricing)	**Affine scaling**
Total # of iterations	(534, 11364)	(38, 71)
Ratio of time per iteration	(0.0141, 0.2947)	1
Total cpu time (secs)	(2.74, 123.62)	(0.85, 43.80)
Ratios of cpu times	(1.52, 5.12)	1

- Promising approach to large real-world LP problems

Copyright ©2004 by K. Pattipati

Summary

- Methods for solving LP problems
  - Revised Simplex method
  - Ellipsoid method….not practical
  - Karmarkar's projective scaling (interior point method)

- Implementation issues of the Least-Squares subproblem of Karmarkar's method ….. More in *Linear Programming and Network Flows* course

- Comparison of Simplex and projective methods

Copyright ©2004 by K. Pattipati