3.3 Sensitivity Analysis

> Analyse the stability of an optimal (primal or dual) solution against the (plus and minus) changes of an coefficient in the LP.

There are two types of analyses that are computationally easy.

A. Stability of an dual optimal solution w.r.t. RHS changes

Fix one constraint, and consider the modified LP where the RHS in the fixed constraint is changed by a parameter α.

$$\begin{array}{lrcrcrcl} \max & 3x_1 & + & 4x_2 & + & 2x_3 & & \\ \text{subject to} & 2x_1 & & & & & \le & 4+\alpha \\ & x_1 & & & + & 2x_3 & \le & 8 \\ & & & 3x_2 & + & x_3 & \le & 6 \\ & x_1 & \ge 0, & x_2 & \ge 0, & x_3 & \ge & 0 \end{array}$$

> Ranges of α for which the dual optimal solution (dual price)
> $$(\bar{y}_1, \bar{y}_2, \bar{y}_3) = (\frac{4}{3}, \frac{1}{3}, \frac{4}{3})$$
> stays optimal?

$$\Longrightarrow \quad \underset{\text{allowable decrease}}{\underline{-4}} \le \alpha \le \underset{\text{allowable increase}}{\underline{12}}$$

Remark 3.1 *The standard technique to be discussed in Section 4.6 is not completely satisfactory: it might return ranges that are not widest possible.*

B. Stability of an optimal solution w.r.t. objective changes

Fix one coefficient of the objective function, and consider the modified LP where the fixed objective coefficient is changed by a parameter β.

$$\begin{array}{lrcrcrcl} \max & 3x_1 & + & (4+\beta)x_2 & + & 2x_3 & & \\ \text{subject to} & 2x_1 & & & & & \le & 4 \\ & x_1 & & & + & 2x_3 & \le & 8 \\ & & & 3x_2 & + & x_3 & \le & 6 \\ & x_1 & \ge 0, & x_2 & \ge 0, & x_3 & \ge & 0 \end{array}$$

Ranges of β for which the optimal solution
$$(\bar{x}_1, \bar{x}_2, \bar{x}_3) = (2, 1, 3)$$
stays optimal?

$$\Longrightarrow \quad \underset{\text{allowable decrease}}{\underset{\|}{\underline{-4}}} \;\leq\; \beta \;\leq\; \underset{\text{allowable increase}}{\underset{\|}{\underline{2}}}$$

Remark 3.2 *The same remark as Remark 3.1 applies to this case.*

Sensitivity Analysis in LINDO

```
   RANGES IN WHICH THE BASIS IS UNCHANGED

                              OBJ COEFFICIENT RANGES
 VARIABLE          CURRENT        ALLOWABLE        ALLOWABLE
                    COEF          INCREASE         DECREASE
       X1         3.000000        INFINITY            2.666667
       X2         4.000000          2.000000          4.000000     beta
       X3         2.000000          5.333333           .666667

                              RIGHTHAND SIDE RANGES
      ROW          CURRENT        ALLOWABLE        ALLOWABLE
                     RHS          INCREASE         DECREASE
        2         4.000000         12.000000          4.000000     alpha
        3         8.000000          6.000000          6.000000
        4         6.000000        INFINITY            3.000000
```

Sensitivity Analysis in CPLEX

```
                                             OBJ Sensitivity Ranges
Variable Name      Reduced Cost          Down          Current             Up
x1                         zero        0.3333           3.0000      +infinity
x2                         zero          zero           4.0000         6.0000
x3                         zero        1.3333           2.0000         7.3333
---
Display what: rhs
Display RHS sensitivity for which constraint(s): c1-c3
                                             RHS Sensitivity Ranges
Constraint Name       Dual Price          Down          Current             Up
c1                       1.3333          zero           4.0000        16.0000
c2                       0.3333        2.0000           8.0000        14.0000
c3                       1.3333        3.0000           6.0000      +infinity
```

Figure 3.2 exhibits geometric meaning of sensitivity analysis. There the supply of gamay grape (the second constraint) is changed gradually from the current 8 tons to 1 ton. The primal optimal solution is indicated by • which moves from the top point on the plane of supply 8 to the bottom point on the supply 2 smoothly, and then turns right to the point on the supply 1. This means the supply 2 which corresponds to the degrease of 6 tons ($\alpha = -6$) is critical. In fact, when the supply becomes smaller than 2, the structure of the feasible region becomes different. In particular the pinot grape constraint $2x_1 \leq 4$ becomes nonbinding (since its supply is too much and cannot be fully used for any feasible productions) and thus its price becomes zero. Therefore the sensitivity analysis returns the allowable decrease as 6.

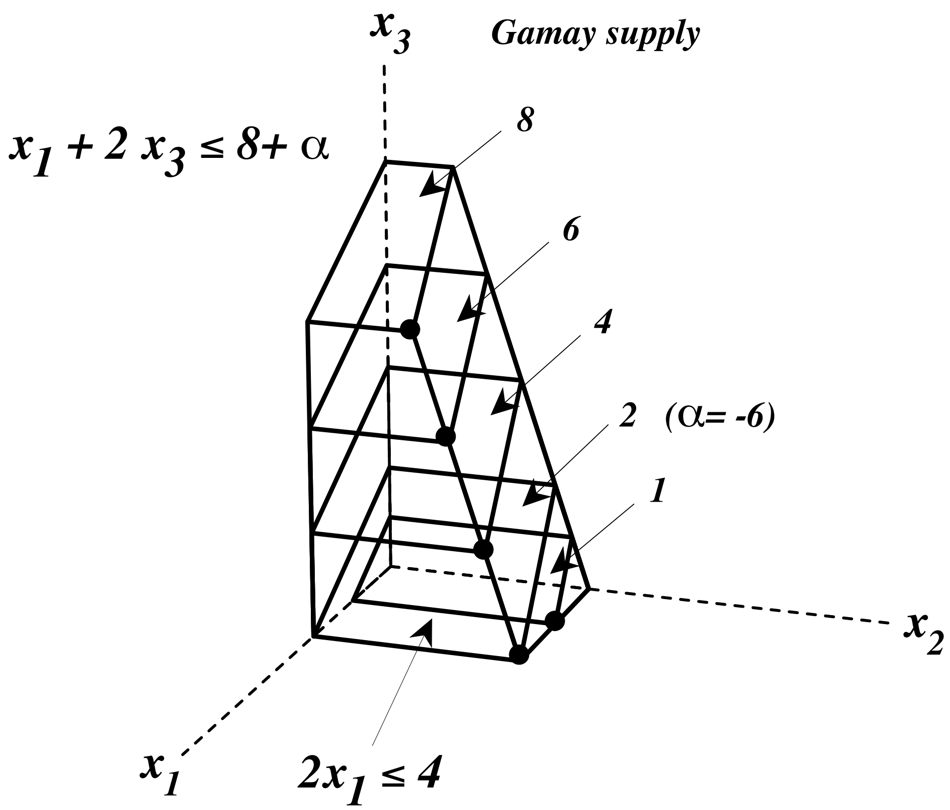


Figure 3.2: Sensitivity of the dual price with respect to changes in Gamay supply

One can easily imagine how the feasible region changes as α increases. The shape smoothly changes as the supply of gamay increases until the constraint hyperplane itself disappears from the feasible region to form a sharp pyramid. This critical α (allowable increase) is 6 (as shown in the ouput of LINDO/CPLEX), and the dual price of gamay becomes 0 for gamay supply larger than 14 ($= 8 + 6$).

Chapter 4

LP Algorithms

The main goal of Chapter 4 is to present finite algorithms for solving a general LP. We present two algorithms, the criss-cross method and the simplex method. The strong duality theorem, Theorem 2.2, and two other theorems, Theorem 2.4 and Theorem 2.5, presented earlier, are direct consequences of their finiteness.

Both algorithms rely on an elementary operation on matrices, called a **pivot operation**. They both maintain a pair of primal solution x and dual solution y which satisfy the complementary slackness conditions in Theorem 2.3. They generate a new solution pair (x, y) using a pivot operation until both solutions become feasible or a certificate for infeasibility of the primal or the dual problem. The simplex method requires an initial primal feasible solution and preserves the primal feasibility of x throughout, whereas the criss-cross method does not require a feasible solution and may not preserve the feasibility even after it attained.

We shall not present interior-point algorithms in this chapter. These algorithms often work for the convex nonlinear program, a larger class of optimization problems, and thus will be given later in Chapter 10. One variation of such algorithms carry a dual pair of **feasible** solutions (x, y), and generate a sequence of new feasible pairs until the complementary slackness condition is "sufficiently" satisfied so that a simple modification of the solutions leads to an optimal solution pair.

To make a clean presentation of the pivot algorithms, we introduce an extended notation for matrices in which the rows and columns of matrices are indexed by finite sets, rather than by consecutive integer sets. The conventional notation imposes annoying unnecessary structures, such as consecutive indices $1, 2, \ldots, n$ for both rows and columns, to mathematical models to be represented by matrices. We believe that the new notation is natural and easy to manipulate for those who studied the basic linear algebra with usual matrix notations.

The immediate benefit of the new notation is simple and precise descriptions of algorithms. They are easily readable by humans and almost readable by computers at the same time. This also makes implementations in high level languages straightforward.

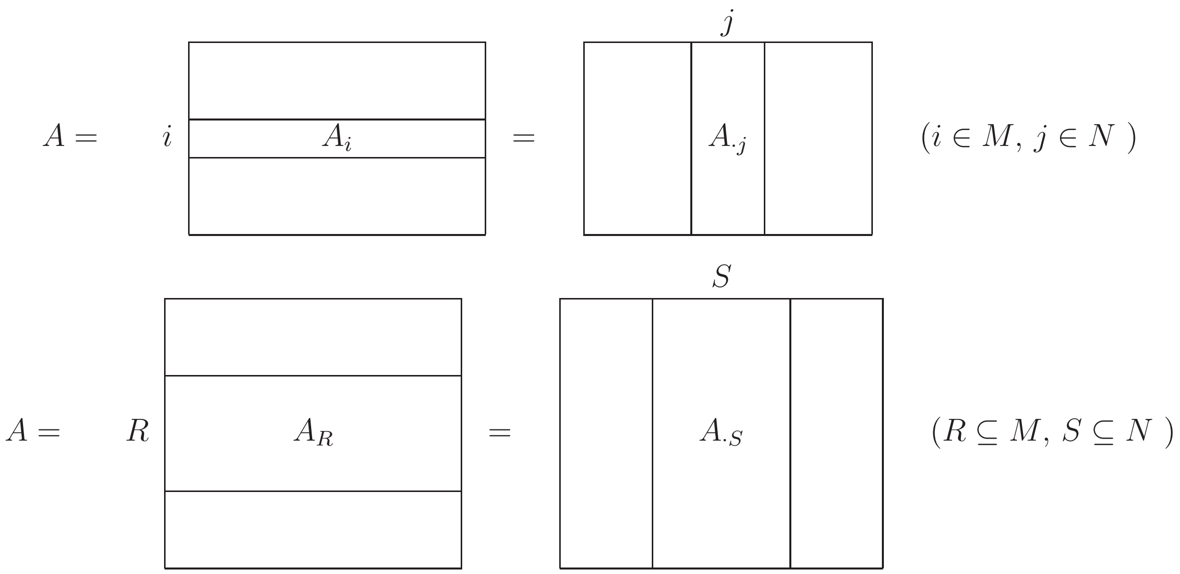
4.1 Matrix Notations

For finite sets M and N, an $M \times N$ *matrix* is a family of doubly indexed numbers or variables

$$A = (a_{ij} : i \in M, j \in N)$$

where each member of M is called a *row index*, each member of N is called a *column index*, and each a_{ij} is called a (i, j)-*component* or (i, j)-*entry*.

For $R \subseteq M$ and $S \subseteq N$, the $R \times S$ matrix $(a_{rs} : r \in R, s \in S)$ is called a submatrix of A, and will be denoted by A_{RS}. We use simplified notations like, A_R for A_{RN}, $A_{\cdot S}$ for A_{MS}, A_i for $A_{\{i\}}$, and $A_{\cdot j}$ for $A_{\cdot\{j\}}$. For each i, j, $A_i = (a_{ij} : j \in N)$ is called the *i-row* and the matrix $A_{\cdot j} = (a_{ij} : i \in M)$ is called the *j-column*. Again, more visually,



$M \times N$ matrices might be simply called *matrices* if the row and column index sets are clear from the context or irrelevant. *Real, rational*, *integer* matrices are matrices whose components take real, rational, integer values, respectively.

For positive integer m, n, an $m \times n$ matrix is an $M \times N$ matrix where M and N are some sets of cardinality m and n. Such $m \times n$ matrices will be used when there is no need to specify the index sets. Unless otherwise specified, an $m \times n$ matrix should be considered as $M \times N$ matrix with the usual index sets $M = \{1, \ldots, m\}$ and $N = \{1, \ldots, n\}$. We shall use mixed forms also: $m \times J$ matrix and $I \times n$ matrix whose meaning should be clear.

The *negative* $-A$ and the scalar multiple αA of a matrix A are defined in the obvious manner. The *transpose* A^T of an $M \times N$ matrix A is the $N \times M$ matrix $B = (b_{ij} : i \in N, j \in M)$ with $b_{ij} = a_{ji}$. The *sum* $A + B$ of $M \times N$ matrices A and B is the $M \times N$ matrix C with $c_{ij} = a_{ij} + b_{ij}$. The *product* $A \times B$ of an $M \times N$ matrix A and a $N \times K$ matrix B is the $M \times K$ matrix C with $c_{ik} = \sum_{j \in N} a_{ij} \times b_{jk}$.

An *identity* matrix is an $K \times K$ matrix A with $a_{ij} = 1$ if $i = j$ and $a_{ij} = 0$ otherwise. The $K \times K$ identity matrix is denoted by $I^{(K)}$. A matrix of all zeros is denoted by $\mathbf{0}$. Its row and column index sets are usually clear from the context.

An $M \times N$ matrix is called a *column M-vector* if the column index set N is a singleton and identified with the set $\{1\}$. Similarly, an $M \times N$ matrix is called a *row N-vector* if the row index set M is a singleton and identified with the set $\{1\}$. We use simple terms *row vectors* and *column vectors* if there is no ambiguity. Unless otherwise specified, a *vector* means a column vector. Row and column vectors are denoted by lower-case letters whose components are singly indexed: $x = (x_j : j \in N)$. In the extreme case when both of the index sets M and N are identified with $\{1\}$, such a matrix is both a row and column vector, and is called a *scalar*.

When we say the *i-row vector* of a matrix A, we mean the row vector obtained from the i-row A_i of A by identifying the only row index i with 1. Similarly we use the term *j-column vector*. The j-column vector of a $K \times K$ identity matrix $I^{(K)}$ is denoted by $e_j^{(K)}$ for $j \in K$ and called a *unit* vector.

The set of all $M \times N$ real matrices is denoted by $R^{M \times N}$, and the set of all real column N-vectors is denoted by R^N.

4.2 LP in Dictionary Form

An LP in canonical form is

$$
\begin{array}{lrl}
\max & c^T x & \\
\text{subject to} & Ax & \le b \\
 & x & \ge \mathbf{0},
\end{array} \tag{4.1}
$$

where A, b and c are given matrix and vectors.

A concrete example we use here is again Chateau ETH Production Problem, Example 1.1:

$$
\begin{array}{lrcrcrcl}
\max & 3x_1 & + & 4x_2 & + & 2x_3 & & \\
\text{subject to} & & & & & & & \\
 & 2x_1 & & & & & \le & 4 \\
 & x_1 & & & + & 2x_3 & \le & 8 \\
 & & & 3x_2 & + & x_3 & \le & 6 \\
 & x_1 & \ge 0, & x_2 & \ge 0, & x_3 & \ge & 0.
\end{array}
$$

To solve such an LP, it is convenient to convert all inequalities to equalities with extra nonnegative variables, called *slack variables*. For example, the constraints can be converted to an equivalent system by adding new variables x_4, x_5 and x_6:

$$
\begin{array}{rcrcrcrcrcrcl}
2x_1 & & & & & + & x_4 & & & & & = & 4 \\
x_1 & & & + & 2x_3 & & & + & x_5 & & & = & 8 \\
 & & 3x_2 & + & x_3 & & & & & + & x_6 & = & 6 \\
x_1 & \ge 0, & x_2 & \ge 0 & x_3 & \ge 0 & x_4 & \ge 0 & x_5 & \ge 0 & x_6 & \ge & 0.
\end{array}
$$

The indices 4, 5, 6 of slack variables represents the original inequality constraints (Pinot, Gamay and Chasselas constraints, respectively) and the value of each slack variable is actually a slack or surplus of the supply, i.e. if x_4 is positive at some feasible solution, the Pinot grape supply 4 is not completely used and x_4 is the surplus.

Also one can set the objective function $c^T x$ as a new variable x_f as

$$-3x_1 - 4x_2 - 2x_3 + x_f = 0.$$

Therefore, the original LP is equivalent to

$$\begin{array}{lrcl}
\text{maximize} & x_f & = & 0 + 3x_1 + 4x_2 + 2x_3 \\
\text{subject to} & x_4 & = & 4 - 2x_1 \\
 & x_5 & = & 8 - x_1 - 2x_3 \\
 & x_6 & = & 6 - 3x_2 - x_3 \\
 & & & x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0
\end{array}$$

Finally we use a new variable x_g to make the equations homogeneous:

$$\begin{array}{lrcl}
\text{maximize} & x_f & = & 0 \cdot x_g + 3x_1 + 4x_2 + 2x_3 \\
\text{subject to} & x_4 & = & 4 \cdot x_g - 2x_1 \\
 & x_5 & = & 8 \cdot x_g - x_1 - 2x_3 \\
 & x_6 & = & 6 \cdot x_g - 3x_2 - x_3 \\
 & & & x_g = 1 \\
 & & & x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0
\end{array}$$

The homogeneous equality system is then written as

$$x_B = D x_N$$

where D is the $B \times N$ matrix with $B = \{f, 4, 5, 6\}$ and $N = \{g, 1, 2, 3\}$:

$$D = \begin{bmatrix} 0 & 3 & 4 & 2 \\ 4 & -2 & 0 & 0 \\ 8 & -1 & 0 & -2 \\ 6 & 0 & -3 & -1 \end{bmatrix}.$$

It is important to note that 0 in the first row and in the first column is not denoted by d_{11} but d_{fg}, and -2 in the second row and in the second column is not denoted by d_{22} but d_{41}. This non-conventional notation represents how the original variables are related to each other, while the conventional notation completely ignores this correspondence and introduces some arbitrary relations between the coincidently consecutively ordered rows and columns.

In general, if an LP is given in canonical form, we can transform it to

$$\begin{array}{ll}
\text{maximize} & x_f \\
\text{subject to} & \\
 & \begin{array}{rcl} x_f & = & 0 \cdot x_g + c^T x_{N_0} \\ x_{B_0} & = & b \cdot x_g - A x_{N_0} \\ x_g & = & 1 \\ x_j \geq 0 & & \forall j \in B_0 \cup N_0 \end{array} .
\end{array}$$

where a new vector x is indexed by $E = B_0 \cup N_0 \cup \{f, g\}$ and the original vector x is replaced by x_{N_0}.

By setting

$$D = \begin{bmatrix} 0 & c^T \\ b & -A \end{bmatrix}. \tag{4.2}$$

the canonical LP is equivalent to an LP in *dictionary form*:

$$\begin{array}{lllll} \text{maximize} & x_f & & & \\ \text{subject to} & & & & \\ & x_B & = & Dx_N & \\ & x_g & = & 1 & \\ & x_j & \geq 0 & \forall j \in E \setminus \{f, g\}. & \end{array} \tag{4.3}$$

where E is a finite set partitioned into two disjoint subsets B and N, D is a given $B \times N$ matrix, $g \in N$, $f \in B$. Here f is called the *objective index*, and g is the *RHS* (right hand side) index. The set B is called a *basis*, N is a *nonbasis* and D is a *dictionary*. Note that the dictionary D together with two elements f and g represent the LP.

The main goal of Chapter 4 is to present finite algorithms to solve a general LP in dictionary form. We present two such algorithms, the criss-cross method and the simplex method. The strong duality theorem is a consequence of their finiteness.

We define few basic terminologies. A *feasible solution* is a vector $x \in R^E$ satisfying all constraints. An *optimal solution* is a feasible solution maximizing x_f among all feasible solutions. An LP is said to be *feasible* if it admits a feasible solution, and *infeasible* otherwise. An *unbounded direction* is a vector $z \in R^E$ satisfying $z_f > 0$, $z_g = 0$ and all constraints except $z_g = 1$.

One can easily see that if an LP is infeasible or there is an unbounded direction, then it cannot have an optimal solution. The fundamental fact for linear programming (Theorem 1.3) is that the converse is true: **an LP has an optimal solution if and only if it is feasible and it admits no unbounded direction**. This is a direct corollary of the duality theorem to be discussed below.

The *dual* problem of an LP in dictionary form is defined as the following LP in dictionary form:

$$\begin{array}{lllll} \text{maximize} & y_g & & & \\ \text{subject to} & & & & \\ & y_N & = & -D^T y_B & \\ & y_f & = & 1 & \\ & y_j & \geq & 0 & \forall j \in E \setminus \{f, g\}. \end{array} \tag{4.4}$$

Thus the basis and the nonbasis are interchanged, f and g are interchanged, and the dictionary is replaced by its negative transpose.

Proposition 4.1 *The definition (4.4) of the dual LP in dictionary form extends the definition (2.6) of the dual LP in canonical form.*

Proof. Let D be the reduction matrix (4.2) of a canonical form LP to a dictionary form LP. Since

$$-D^T = \begin{bmatrix} 0 & -b^T \\ -c & A^T \end{bmatrix}.$$

the dual LP of the primal LP given by D is

(4.5)
$$\begin{array}{lrcl} \text{maximize} & y_g & & \\ \text{subject to} & & & \\ & y_g & = & 0 \cdot y_f - b^T y_{B_0} \\ & y_{N_0} & = & - c \cdot y_f + A^T y_{B_0} \\ & y_f & = & 1 \\ & y_j \geq 0 & & \forall j \in B_0 \cup N_0 \quad . \end{array}$$

We can then substitute y_f with 1 and remove y_{N_0} to get an equivalent LP:

(4.6)
$$\begin{array}{lrl} \min & b^T y_{B_0} & \\ \text{subject to} & A^T y_{B_0} & \geq c \\ & y_{B_0} & \geq \mathbf{0}. \end{array}$$

This problem is exactly the dual of the LP in canonical form defined in (2.6) if we consider the variables y_{B_0} as y. ∎

We remark two important things with the proof above. First, the dual objective function y_g in the dictionary form (4.5) represents $-b^T y$ and this explains why the dual is also maximization in dictionary form. Secondly, the variable y_{B_0} in the reduced dual LP (4.6) in canonical form is indexed by the row index set B_0 of A. Of course this is natural because the dual variables correspond to the primal constraints associated with the rows of A. In the concrete example above, $B_0 = \{4, 5, 6\}$ and thus

(4.7)
$$\begin{array}{lrcrcrcr} \min & 4y_4 & + & 8y_5 & + & 6y_6 & & \\ \text{subject to} & 2y_4 & + & y_5 & & & \geq & 3 \\ & & & & & 3y_6 & \geq & 4 \\ & & & 2y_5 & + & y_6 & \geq & 2 \\ & y_4 & \geq 0, & y_5 & \geq 0, & y_6 & \geq & 0 \end{array}$$

This LP is equivalent to the dual LP in Example 2.1 whose variables are y_1, y_2, y_3 rather than y_4, y_5, y_6. The form of Example 2.1 is widely used and considered standard. Yet our form (4.7) is more appropriate theoretically, since the indices 1, 2, 3 represent the columns of A (red, white, rose wines), where as 4, 5, 6 represent the different objects (Pinot, Gamay, Chasselas grapes), the rows of A, and they correspond to the primal variables x_4, x_5, x_6,

One simple but important observation for a dual pair of LPs in dictionary format is that

(4.8)
$$x^T y = x_B^T\, y_B + x_N^T\, y_N = 0$$

for any primal and dual feasible solutions x and y. In fact this orthogonality of primal and dual vectors is true as long as they satisfy the dictionary equalities: $x_B = Dx_N$ and $y_N = -D^T y_B$.

For any fixed $j \in N$, by setting $\overline{x}_j = 1$, $\overline{x}_{N-j} = \mathbf{0}$, and $\overline{x}_B = D_{\cdot j}$ for the primal, we obtain a vector, denoted by $x(B, j)$, satisfying the dictionary equality system. In particular, the vector $x(B, g)$, satisfying all the equalities including $x_g = 1$, is called the *(primal) basic solution* with respect to B. The remaining vectors $x(B, j)$ for $j \in N - g$ have zero g component and are called basic directions.

Similarly, for any fixed $i \in B$, by setting $\overline{y}_i = 1$, $\overline{y}_{B-i} = \mathbf{0}$, and $\overline{y}_N = -(D_i)^T$ for the dual, we obtain a vector, denoted by $y(B, i)$ satisfying all dual dictionary equality system. The vector $y(B, f)$ is called the *dual basic solution* and vectors $y(B, i)$ for $i \in B - f$ are the *dual basic directions*.

Basic Solutions and Basic Directions (Reminder)

The basic vector $x(B, j)$ for $j \in N$ is
the unique solution $\overline{x}$ to $x_B = Dx_N$ such that

$$\begin{aligned} &\overline{x}_j = 1 \\ &\overline{x}_{N-j} = \mathbf{0} \\ &\overline{x}_B = D_{\cdot j} \quad . \end{aligned}$$

$$\begin{aligned} x(B, g) &: \quad \textit{the basic solution w.r.t. } B \\ x(B, j) &: \quad \textit{the basic direction w.r.t. } B \text{ and } j \in N - g \end{aligned}$$

The dual basic vector $y(B, i)$ for $i \in B$ is
the unique solution $\overline{y}$ to $y_N = -D^T y_B$ such that

$$\begin{aligned} &\overline{y}_i = 1 \\ &\overline{y}_{B-i} = \mathbf{0} \\ &\overline{y}_N = -(D_i)^T \quad . \end{aligned}$$

$$\begin{aligned} y(B, f) &: \quad \textit{the dual basic solution w.r.t. } B \\ y(B, i) &: \quad \textit{the dual basic direction w.r.t. } B \text{ and } i \in B - f \end{aligned}$$

We state the weak and the strong duality theorems for LPs in dictionary form:

Theorem 4.2 (LP Weak Duality) *For any LP in dictionary form (4.3) and for any primal and dual feasible solutions x and y, $x_f + y_g \leq 0$.*

Proof. Let x and y be primal and dual feasible solutions. Since $x^T y = 0$ as remarked in

(4.8),

$$
\begin{aligned}
x_f + y_g &= x_f y_f + x_g y_g && (\because x_g = 1 \text{ and } y_f = 1) \\
&= x^T y - \sum_{j \in E \setminus \{f,g\}} x_j y_j \\
&= - \sum_{j \in E \setminus \{f,g\}} x_j y_j \\
&\le 0. && (\because x_j \ge 0 \text{ and } y_j \ge 0 \text{ for all } j \in E \setminus \{f,g\})
\end{aligned}
$$

■

One corollary of the weak duality is: if the primal LP is unbounded, then the dual LP is infeasible. This is true because if the primal LP is unbounded and y is dual feasible, there is a primal feasible solution x with x_f as large as one wants, for example $1 - y_g$, for which $x_f + y_g \le 0$ fails. The other consequence is: if a primal feasible solution x and a dual feasible solution y satisfy $x_f + y_g = 0$ then both of them are optimal. The strong duality theorem below shows that one can always verify the optimality of a feasible solution by exhibiting a dual feasible solution satisfying this equality.

Theorem 4.3 (LP Strong Duality) *For any LP in dictionary form (4.3), the following statements hold:*

(a) *If the primal and dual LPs are both feasible then both have optimal solutions. Furthermore their optimal values sum up to zero.*

(b) *If one of the primal or dual LPs is infeasible then neither of the LPs has an optimal solution. Moreover, if the dual (primal) LP is infeasible, the primal (dual, respectively) LP is either infeasible or unbounded.*

We shall prove this theorem in Section 4.4.

The dictionary form LP (4.1) is ideal for presenting certain algorithms for solving an LP. Such algorithms use pivot operations to replace the current basis B with a new basis $B - r + s$ for some $r \in B$ and $s \in N$ and to update the dictionary D so that the associated linear equality system stays equivalent (and thus the LP stays equivalent). We keep pivoting until the associated basic solution is optimal or we find certificates for the LP having no optimal solution. Dictionary facilitates simple certificates for optimality, infeasibility, unboundedness, etc.

We define four different types of dictionaries (and bases). A dictionary D (or the associated basis B) is called *(primal) feasible* if $d_{ig} \ge 0$ for all $i \in B - f$. A dictionary D (or the associated basis B) is called *dual feasible* if $d_{fj} \le 0$ for all $j \in N - g$. A dictionary D (or the associated basis B) is called *(primal) inconsistent* if there exists $r \in B - f$ such that $d_{rg} < 0$ and $d_{rj} \le 0$ for all $j \in N - g$. A dictionary D (or the associated basis B) is called *dual inconsistent* if there exists $s \in N - g$ such that $d_{fs} > 0$ and $d_{is} \ge 0$ for all $i \in B - f$.

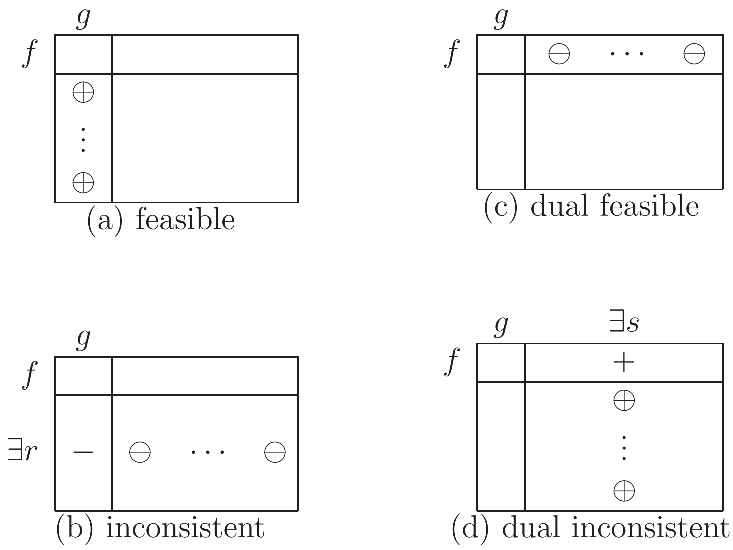


Figure 4.1: Four types of dictionaries for LP

The four types of dictionaries are illustrated in Figure 4.1. Note that the symbols $+$, $-$, $\oplus$ and $\ominus$ denote positive, negative, nonnegative and nonpositive entries.

Proposition 4.4 *For any LP in dictionary form, the following statements hold.*

(a) *If the dictionary is feasible then the associated basic solution is feasible.*

(b) *If the dictionary is dual feasible then the associated dual solution is feasible.*

(c) *If the dictionary is both primal and dual feasible then the associated basic solution $\overline{x}$ and the associated dual basic solution $\overline{y}$ are optimal, and furthermore $\overline{x}_f + \overline{y}_g = 0$.*

Proof.

(a) If the dictionary is feasible, the basic solution $x(B, g)$ satisfies all the nonnegativity conditions and thus is feasible.

(b) If the dictionary is dual feasible, the dual basic solution $y(B, f)$ satisfies all the nonnegativity conditions and thus is dual feasible.

(c) Assume the dictionary is both primal and dual feasible. Since $x(B, g)_f = d_{fg}$ and $y(B, f)_g = -d_{fg}$, they add up to zero. By the Weak Duality, Theorem 4.2, they are both optimal solutions.

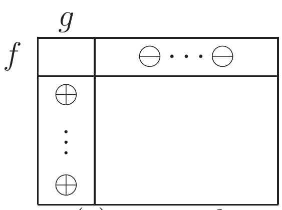
■

One might profit from a direct way to see the optimality of $x(B, g)$ when the dictionary is both feasible and dual feasible (Statement (c) above). Observe the equality constraint:

$$x_f = d_{fg}x_g + \sum_{j \in N-g} d_{fj}x_j$$

that must be satisfied by any primal feasible solution. The basic solution is feasible and attains the objective value d_{fg}. By the dual feasibility $d_{fj} \leq 0$ for all $j \in N - g$, $x_f \leq d_{fg}$ for any feasible solution x, since any feasible x must satisfy $x_j \geq 0$ for any $j \in N - g$.

Because of the statement (c) above, a dictionary D (or the associated basis B) is called *optimal* if it is both primal and dual feasible, see Figure 4.2.



(e) optimal

Figure 4.2: Optimal dictionary

Proposition 4.5 *For any LP in dictionary form, the following statements hold.*

(a) *If the dictionary is inconsistent then the LP is infeasible and the dual LP has an unbounded direction.*

(b) *If the dictionary is dual inconsistent then the dual LP is infeasible and the primal LP has an unbounded direction.*

Proof.

(a) Suppose the dictionary is inconsistent. Observe the row r indicating the inconsistency:

$$x_r = d_{rg} x_g + \sum_{j \in N - g} d_{rj} x_j$$

that must be satisfied by any primal feasible solution. Since $d_{rg} < 0$ and $d_{rj} \leq 0$ for all $j \in N - g$, $x_r < 0$ for any x satisfying this equality, $x_g = 1$ and $x_j \geq 0$ for all $j \in N - g$. Therefore the LP has no feasible solution.

Now let w be the dual basic direction $y(B, r)$. Then w satisfies all dual constraints except for $w_f = 1$, and we have $w_g > 0$ and $w_f = 0$. Thus w is a dual unbounded direction.

(b) The proof is similar. Observe the corresponding dual constraint.

∎

4.3 Pivot Operation

In this section, we present two finite algorithms for LP. Both algorithms transform an LP in dictionary form to an equivalent LP in dictionary form with a modified dictionary using an elementary matrix operation. This operation is called a pivot operation.

Suppose we have an LP in dictionary form (4.3), and consider the dictionary equality:

$$x_B = Dx_N \tag{4.9}$$

where D is a $B \times N$ matrix, and $(B,\, N)$ is a partition of E. Denote the set of solutions to the system by

$$V(D) := \{x \in R^E : x_B = Dx_N\}. \tag{4.10}$$

Now suppose that $r \in B$, $s \in N$, and the (r, s)-entry d_{rs} of D is nonzero. Then one can transform the equality system (4.9) to an equivalent system with the new basis $B' = B-r+s$ and the new nonbasis $N' = N - s + r$ by simple substitution:

- First solve the r-th equation in (4.9):

$$x_r = \sum_{j \in N} d_{rj} x_j$$

  in terms of x_s to obtain:

$$x_s = \sum_{j \in N-s} -\frac{d_{rj}}{d_{rs}} x_j + \frac{1}{d_{rs}} x_r. \tag{4.11}$$

- Then eliminate x_s using this equation in every other equations in (4.9) to get:

$$\begin{aligned} x_i &= \sum_{j \in N} d_{ij} x_j \\ &= \sum_{j \in N-s} d_{ij} x_j + d_{is} x_s \\ &= \sum_{j \in N-s} d_{ij} x_j + d_{is} \left(\sum_{j \in N-s} -\frac{d_{rj}}{d_{rs}} x_j + \frac{1}{d_{rs}} x_r \right) \\ &= \sum_{j \in N-s} \left(d_{ij} - \frac{d_{is} \cdot d_{rj}}{d_{rs}} \right) x_j + \frac{d_{is}}{d_{rs}} x_r \quad (\text{ for all } i \in B - r). \end{aligned} \tag{4.12}$$

- To summarize, by defining the $B' \times N'$ matrix $D' = [d'_{ij}]$ by

$$d'_{ij} = \begin{cases} \frac{1}{d_{rs}} & \text{if } i = s \text{ and } j = r \\ -\frac{d_{rj}}{d_{rs}} & \text{if } i = s \text{ and } j \neq r \\ \frac{d_{is}}{d_{rs}} & \text{if } i \neq s \text{ and } j = r \\ d_{ij} - \frac{d_{is} \cdot d_{rj}}{d_{rs}} & \text{if } i \neq s \text{ and } j \neq r \end{cases} \quad (i \in B' \text{ and } j \in N'), \tag{4.13}$$

we obtain a new system of equations that is equivalent to the original system (4.9):

$$x_{B'} = D' x_{N'}. \tag{4.14}$$

We say that the matrix D' is obtained by a *pivot operation* in D on the position (r, s). See Figure 4.3 for a schematic description.

$$D = \begin{array}{c|ccccc|} & & j & & s & \\ \hline & & \vdots & & \vdots & \\ i & \cdots & d_{ij} & \cdots & d_{is} & \cdots \\ & & \vdots & & \vdots & \\ r & \cdots & d_{rj} & \cdots & d_{rs} & \cdots \\ & & \vdots & & \vdots & \\ \hline \end{array} \qquad (i \in B - r,\ j \in N - s)$$

$$\Downarrow \text{ Pivot on } (r, s)$$

$$D' = \begin{array}{c|ccccc|} & & j & & r & \\ \hline & & \vdots & & \vdots & \\ i & \cdots & d_{ij} - \dfrac{d_{is} \cdot d_{rj}}{d_{rs}} & \cdots & \dfrac{d_{is}}{d_{rs}} & \cdots \\ & & \vdots & & \vdots & \\ s & \cdots & -\dfrac{d_{rj}}{d_{rs}} & \cdots & \dfrac{1}{d_{rs}} & \cdots \\ & & \vdots & & \vdots & \\ \hline \end{array} \qquad (i \in B - r,\ j \in N - s)$$

Figure 4.3: Pivot Operation on (r, s)

We state the most important facts on pivot operations.

Proposition 4.6 *Suppose D' is obtained by a pivot operation in D on the position (r, s). Then the following statements hold:*

(a) *(Reversibility) The original dictionary D is obtained from D' by a pivot operation on (s, r).*

(b) *(Equivalence) The associated equation systems are equivalent, that is, the set of solutions are identical $V(D) = V(D')$.*

(c) *(Dual Equivalence) The associated dual equation systems are equivalent, that is, $V(-D^T) = V(-D'^T)$.*

Proof.

(a) This follows from the definition with straightforward calculation.

(b) By the construction, every solution of the original system for D satisfies the new system, i.e., $V(D) \subseteq V(D')$. The converse relation follows from (a).

(c) Let $U = -D^T$. Since $u_{sr} = -d_{rs} \neq 0$, we can perform a pivot operation in U on (s, r) to obtain a new matrix U'. It is easy to see that $U' = -D'^T$. By (b), we have $V(U) = V(U')$ and this proves (c).

∎

Finally, we note that the pivot operation can be interpreted in terms of matrix multiplications. The dictionary system (4.9)

$$x_B = Dx_N$$

can be written as

$$[I^{(B)} \quad -D] \begin{bmatrix} x_B \\ x_N \end{bmatrix} = \mathbf{0}. \tag{4.15}$$

By setting $A = [I^{(B)} \quad -D]$ ($B \times E$ matrix), the system is

$$A\,x = \mathbf{0}.$$

The large matrix A is known as the *tableau* associated with the dictionary system. One can represent the dictionary system by tableaus or dictionaries[1].

Now suppose $a_{rs}(= -d_{rs}) \neq 0$. Setting $B' = B - r + s$ and $N' = N + r - s$, it can be shown (how?) that the matrix $A_{\cdot B'}$ has the left inverse, i.e. a unique $B' \times B$ matrix T such that $T\,A_{\cdot B'} = I^{(B')}$. We denote this matrix by $A^{-1}_{\cdot B'}$. By multiplying this matrix from the left of (4.15), we obtain a new equivalent system:

$$A^{-1}_{\cdot B'} A\,x = \mathbf{0}. \tag{4.16}$$

Since

$$A^{-1}_{\cdot B'} A = [I^{(B')} \quad A^{-1}_{\cdot B'} A_{\cdot N'}] \tag{4.17}$$

the new dictionary D' obtained by a pivot on (r, s) can be considered as

$$D' = -A^{-1}_{\cdot B'} A_{\cdot N'}. \tag{4.18}$$

4.4 Pivot Algorithms and Constructive Proofs

We shall call an LP dictionary D *terminal* if it is optimal, inconsistent or dual inconsistent. In this section, we present finite algorithms that transforms any dictionary to a terminal dictionary.

[1]Our preference in this note is dictionary since it is more compact and it represents LP duality nicely.

The first one, called the (least-index) *criss-cross* method is one of the simplest pivot algorithms. In fact it is almost trivial to code this algorithm. However it is in general quite inefficient for large-scale problems. The second pivot method, called the *simplex method*, is extremely practical but its description requires more space and care. The simplex method has been implemented in many commercial codes and used to solve very large real-world problems.

From the theoretical point of view, there is no polynomial-time pivot algorithm known for linear programming. This means the number of pivot operations needed to solve an LP of input size L might grow exponentially in L for both algorithms we present here. Presently the known polynomial-time algorithms employ methods developed for non-linear programming, such as ellipsoid methods and interior-point methods.

4.4.1 The Criss-Cross Method and a Proof of Strong Duality

The criss-cross method uses two types of pivot operations, both of which are quite natural. For a $B \times N$ dictionary D with $d_{rs} \neq 0$, a pivot on (r, s) is said to be *admissible* if one of the following two conditions is satisfied (see Figure 4.4):

(I) $d_{rg} < 0$ and $d_{rs} > 0$;

(II) $d_{fs} > 0$ and $d_{rs} < 0$.

In the first case (I), the current dictionary is not feasible because $d_{rg} < 0$, but after the pivot the new entry (d'_{sg}) will become positive. For the second case (II), the dictionary is not dual feasible because $d_{fs} > 0$, but after the pivot the corresponding entry will have a correct (negative) sign. Furthermore, if a dictionary is not terminal, there is always an admissible pivot.

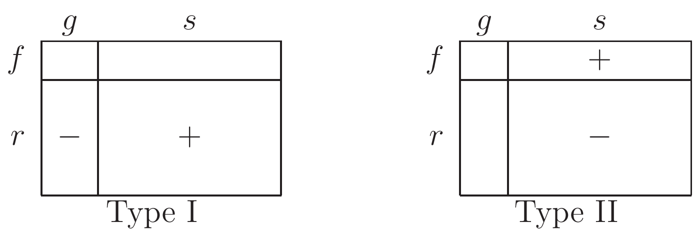


Figure 4.4: Admissible Pivots

The (least-index) criss-cross method given below starts with any $B \times N$ dictionary D with two specified elements $f \in B$ and $g \in N$, and apply admissible pivots as long as the dictionary is not terminal. Note $E = B \cup N$ denotes the set of all variable indices.

```
procedure CrissCross(D,g,f);
begin
    fix a linear order on E \ {f, g};
    status:="unknown";
    while status="unknown" do
        k := min({i ∈ B − f : d_ig < 0} ∪ {j ∈ N − g : d_fj > 0});
        if no such k exists then
            status:="optimal";
        else
            if k ∈ B then
                r := k; s := min{j ∈ N − g : d_rj > 0};
                if no such s exists then
                    status="inconsistent";
                endif
            else /* k ∈ N */
                s := k; r := min{i ∈ B − f : d_is < 0};
                if no such r exists then
                    status:="dual inconsistent";
                endif;
            endif;
            if status = "unknown" then
                make a pivot on (r, s);
                replace D, B, N with the new D', B', N';
            endif;
        endif;
    endwhile;
    output (status, D); /* D is terminal */
end.
```

Theorem 4.7 (Finiteness) *The criss-cross method terminates after a finite number of pivot operations.*

Before proving this theorem, first observe the consequence.

Corollary 4.8 (Strong Duality Theorem in Dictionary Form) *The dictionary of any LP in dictionary form can be transformed by a finite sequence of pivot operations to a dictionary that is optimal, inconsistent or dual inconsistent.*

This corollary implies the strong duality theorem, Theorem 4.3. First of all, if the LP admits an optimal solution, then both the primal and the dual LPs have optimal solutions whose objective values sum up to zero by Proposition 4.4. If the LP is infeasible, then by the corollary the LP admits an inconsistent or dual inconsistent dictionary. In either case, the dual problem is either infeasible or unbounded by Proposition 4.5.

Directly or indirectly most (if not all) finiteness proofs of the least-index criss-cross method rely on some fundamental proposition on "almost terminal" dictionaries. First we present this proposition and a simple proof.

A dictionary is called *almost terminal* (*with respect to a constraint index* $k \in E \setminus \{f, g\}$) if it is not terminal but by discarding a single row or column (indexed by k) it becomes terminal. Observe that we have four structurally different almost terminal dictionaries, see Figure 4.5.

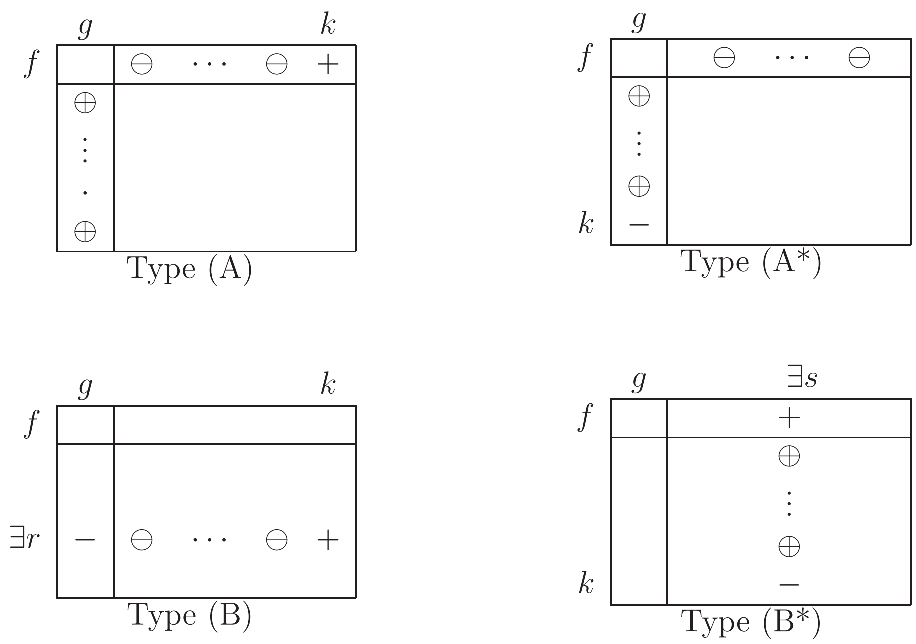


Figure 4.5: Four types of almost terminal dictionaries

Lemma 4.9 *Consider an LP in dictionary form and fix an index* $k \in E \setminus \{f, g\}$. *Then the LP admits (by pivot operations) at most one type of the four almost terminal dictionaries* **A**, **B**, **A*** *and* **B*** *with respect to* k.

Proof. We first transform a given LP to another LP (denoted by LP') by substitution of x_k by its negative $-x_k$. The four types of dictionaries **A**, **B**, **A*** and **B*** for the original LP can be translated to the modified types **a**, **b**, **a*** and **b*** in Figure 4.6 for LP', since the substition affects as the reversal of signs in the row or the column associated with k.

We will show that no two different types from **a**, **b**, **a*** and **b*** can coexist for LP', which will then imply the lemma. There are six pairs of cases to consider. We denote by **(a a*)** to mean the case that both types **a** and **a*** exist simultaneously, etc.

First of all, the types **a** and types **a*** are both optimal and thus indicate both the primal and the dual LP's are feasible. On the other hand the type **b** is an inconstent dictionary indicating the infeasibility of the primal and the type (b*) is a dual inconsistent dictionary indicating the dual infeasibility. Therefore all cases except **(a a*)** and **(b b*)** are impossible.

Consider the case **(b b*)**, and let D, D' be dictionaries of type **b** and **b***, respectively. Now look at the dual pair of directions: $z = x(B', s)$ and $w = y(B, r)$. By the definitions,

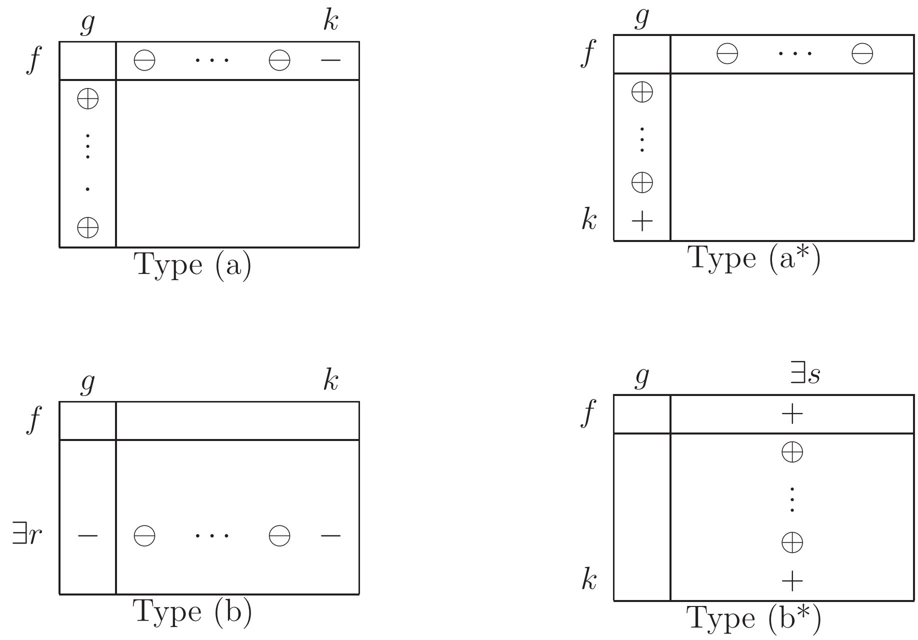


Figure 4.6: Reduced four types

$z_k > 0$ and $w_k > 0$, and their remaining components are all nonnegative. This means $z^T w > 0$, contradicting the orthogonality of the spaces $V(D)$ and $V(-D^T) = V(-(D')^T)$.

To prove that the remaining case **(a a*)** is impossible, assume we have a dictionary D of type **a** and a dictionary D' of type **a***. Let x and x' be the basic solutions associated with B and B', respectively. As we remarked,

(*) both x and x' are optimal for LP'.

Moreover, observing the equation represented by the f-row of D, since $d_{fk} < 0$ and $d_{fj} \leq 0$ for all $j \in N - k$, every feasible solution to LP' with a positive k-component has an objective value strictly less than the optimal value d_{fg}. Thus no optimal solution to LP' can have a positive kth component. Since $x'_k > 0$, we have a contradiction. ∎

Now we prove the finiteness of the criss-cross method, Theorem 4.7.

Proof. (of Theorem 4.7) Since the number of bases is finite, it is enough to show that each basis can be visited at most once by the algorithm. The proof is by contradiction. Assume that a basis is visited twice. Then the algorithm generates a cycle, i.e. starting from this basis, after a certain number of steps the same basis is reached again. Without loss of generality we may also assume that this cycling example is smallest possible in terms of number of variables, and let $E = \{1, \ldots, k\} \cup \{f, g\}$. This implies each variable in $E \setminus \{f, g\}$, in particular the largest index k, must enter and leave a basis during the cycle. (If there is any variable index j which stays in basis or nonbasis in the cycle, one can reduce the size of the cycling example.)

There are two situations when k as a nonbasic element might enter a basis. The first case **A** is when the current dictionary is almost optimal with respect to k, and the second case **B** is when some basic variable, say r, is primal infeasible and d_{rk} determines the only admissible pivot in row r. Similarly, there are only two cases in which k as a basic element might leave a basis. The first case **A*** is when the current dictionary is almost optimal with respect to k, and the second case **B** is when some nonbasic variable, say s, is dual infeasible and d_{ks} determines the only admissible pivot in column s. Thus at least one case of the four possible combinations **(A A*)**, **(A B*)**, **(B A*)** and **(B B*)** must occur. On the other hand, the described cases **A**, **B**, **A*** and **B*** coincide with the four almost terminal dictionaries of Figure 4.5. By Lemma 4.9, none of **(A A*)**, **(A B*)**, **(B A*)** and **(B B*)** can occur, a contradiction. ∎

4.4.2 Feasibility and Farkas' Lemma

In this section, we prove the following feasibility theorem which is almost immediate from the proof of the strong duality theorem in the previous section.

Theorem 4.10 (Feasibility Theorem in Dictionary Form) *The dictionary of any LP in dictionary form can be transformed by a finite sequence of pivot operations to a dictionary that is either feasible or inconsistent.*

Proof. The proof uses the criss-cross method to a modified LP. For a given LP with dictionary D, we replace the objective row D_f by the zero vector $\mathbf{0}^T$. By applying the criss-cross method to this problem, we will get a terminal dictionary, say D', by Corollary 4.8. This dictionary cannot be dual inconsistent, since the dual problem is feasible. This means we obtain either feasible or inconsistent dictionary. Clearly the same pivot sequence applied to the original LP generates the same basis and a dictionary that is feasible or inconsistent. ∎

Corollary 4.11 (Feasibility Theorem in Symmetric Form) *For any matrix $D \in R^{B\times N}$ with $B \cap N = \emptyset$, and for $g \in N$, exactly one of the following statements holds:*

(a) $\exists x \in R^{B\cup N}$ *such that* $x \geq \mathbf{0}$, $x_g > 0$ *and* $x_B = Dx_N$.

(b) $\exists y \in R^{B\cup N}$ *such that* $y \geq \mathbf{0}$, $y_g > 0$ *and* $y_N = -D^T y_B$.

Proof. Consider the LP with $(B + f) \times N$ dictionary $\bar{D} = \begin{bmatrix}\mathbf{0}^T \\ D\end{bmatrix}$ with the first row as the objective row f. By Theorem 4.10, there is either a feasible dictionary or inconsistent dictionary. If it is the first case, we have a feasible solution and thus (a). Otherwise, by Proposition 4.5 (a), we have a dual unbounded direction y, i.e. $y_N = -\bar{D}^T y_{B+f}$, $y_f = 0$, $y \geq 0$ and $y_g > 0$. This y without f component satisfies (b). It is clear that both (a) and (b) cannot hold simultaneously. ∎

Corollary 4.12 (Farkas' Lemma I) *For any matrix $A \in R^{B\times N}$ and any vector $b \in R^B$, exactly one of the following statements holds:*

(a) $\exists x \in R^N$ *such that* $Ax \leq b$ *and* $x \geq \mathbf{0}$.

(b) $\exists \lambda \in R^B$ *such that* $A^T\lambda \geq \mathbf{0}$, $\lambda \geq \mathbf{0}$ *and* $b^T\lambda < 0$.

Proof. Let $D = [b \;\; -A]$. The claim follows from Corollary 4.11. ∎

Corollary 4.13 (Farkas' Lemma II) *For any matrix $A \in R^{B\times N}$ and any vector $b \in R^B$, exactly one of the following statements holds:*

(a) $\exists x \in R^N$ *such that* $Ax = b$ *and* $x \geq \mathbf{0}$.

(b) $\exists \lambda \in R^B$ *such that* $A^T\lambda \geq \mathbf{0}$ *and* $b^T\lambda < 0$.

Proof. Exercise. ∎

4.4.3 The Simplex Method

Unlike the criss-cross method which can start with any basis, the simplex method needs a starting feasible basis. Because of this, it is often categorized as a *two phase method*. The first phase, *Phase I*, is a procedure to find a feasible basis, and the second phase, *Phase II*, starts with a feasible basis and finds a terminal basis.

First we present Phase II, since Phase I can be considered as an application of Phase II to an artificially created problem for finding a feasible basis.

Phase II

The simplex method relies on one type of pivot operation. Let D be a $B \times N$ feasible dictionary D. This means of course $d_{ig} \geq 0$ for all $i \in B - f$. Consider the system $x_B = Dx_N$ with x_g fixed to 1 in detail:

$$x_f = d_{fg} + \sum_{j \in N-g} d_{fj} x_j \tag{4.19}$$

$$x_i = d_{ig} + \sum_{j \in N-g} d_{ij} x_j \qquad \text{(for all } i \in B - f). \tag{4.20}$$

If the basis is optimal we have nothing to do: the current basic solution $\overline{x}$ is optimal. Otherwise there is some nonbasic index $s \in N - g$ with $d_{fs} > 0$. The current basic solution fixes all nonbasic variables indexed by $N - g$ to zero. Consider increasing x_s by some positive value ϵ, while not modifying other nonbasic variables. The basic variables are determined uniquely by the equations (4.19) and (4.20):

$$x(\epsilon)_f = d_{fg} + d_{fs}\epsilon \tag{4.21}$$

$$x(\epsilon)_i = d_{ig} + d_{is}\epsilon \qquad \text{(for all } i \in B - f). \tag{4.22}$$

The objective value $x(\epsilon)_f$ of this new solution increases by $d_{fs} \cdot \epsilon$. Thus as long as the new solution $x(\epsilon)$ stay feasible, this is a better feasible solution. And the feasibility is maintained as long as $x(\epsilon)_i \geq 0$ for all $i \in B - f$. The simplex method uses the largest ϵ that forces some of the basic variables to become zero and one of them to leave the basis.

A pivot on (r, s) is called a *simplex* pivot if

(a) $d_{rs} < 0$;

(b) $-d_{rg}/d_{rs} = \min\{-d_{ig}/d_{is} : i \in B - f \text{ and } d_{is} < 0\}$.

A simple calculation shows that a simplex pivot preserves feasibility. This pivot exists as long as the dictionary is feasible but not terminal. Here is the formal description of the simplex method. The starting dictionary D must be feasible.

```
procedure SimplexPhaseII(D,g,f);
begin
    status:="unknown";
    while status="unknown" do
        Let S = {j ∈ N − g : d_fj > 0};
        if S = ∅ then
            status:="optimal";
        else
(L1)        select any s ∈ S;
            Let R := {i ∈ B − f : d_is < 0};
            if R = ∅ then
                status:="unbounded";
            else
(L2)            make a simplex pivot on (r, s) for any r ∈ R;
                replace D, B, N with the new D', B', N';
            endif;
        endif;
    endwhile;
    output (status, D); /* D is terminal */
end.
```

Note that a simplex pivot may not modify the feasible solution itself. This happens exactly when $d_{rg} = 0$. In general, a dictionary or its basis is called *degenerate* if $d_{rg} = 0$ for some $r \in B - f$. In fact, because of this, the simplex method may not terminate. See Example 4.10.1 for an example for which the simplex method generates an (infinite) cycle of pivots.

There are different ways to make the simplex method terminate in a finite number of pivots. Two standard techniques are lexicographic rule and Bland's rule. Both rules are additional rules to restrict the selection of simplex pivots in (L1) and (L2) so as to avoid cycling. Since Bland's rule is simpler to explain, we present it here. We shall present the lexicographic rule later.

The idea of Bland's rule is to use a linear ordering on the index set and to break ties with the minimum-index rule, just like the least-index criss-cross method selects pivots.

Definition 4.14 *[Bland's Rule or Smallest Index Rule]*

(a) *Fix any linear ordering on* $E \backslash \{f, g\}$*;*

(b) *In (L1), select the smallest* s *from* S*;*

(c) *In (L2), select the smallest* r *among all simplex pivot rows on column* s*.*

Theorem 4.15 *The simplex method (Phase II) with Bland's rule is a finite algorithm.*

Proof. Since the number of bases is finite, it is enough to show that each basis can be visited at most once by the algorithm. The proof is by contradiction. Assume that a basis is visited twice. Then the algorithm generates a cycle, i.e. starting from this basis, after a certain number of steps the same basis is reached again. Without loss of generality we may also assume that this cycling example is minimal and $E = \{1, \ldots, k\} \cup \{f, g\}$. This implies each variable in $E \setminus \{f, g\}$, in particular k, must enter and leave a basis during the cycle.

When k as a nonbasic element enters a basis, the dictionary, say D, is almost optimal except for d_{fk} has a wrong sign (i.e. positive). This is the situation, Type (A) in Figure 4.5. On the other hand, when k as a basic element leaves a basis, the dictionary, say D', is almost dual inconsistent except for d'_{ks} has a wrong sign (i.e., negative) in some selected s column. This is the situation, Type (B*) in Figure 4.5. By Lemma 4.9, these two types of dictionaries cannot coexist, a contradiction. ∎

See Section 4.10.5 for the cycling example solved by the simplex method with Bland's rule.

Phase I

Now we describe the Phase I of the simplex algorithm. If the initial dictionary of a given LP (4.3) is not feasible, one makes an auxiliary LP by adding an auxiliary variable and setting an artificial objective $x_{f'}$ in the original LP:

$$
\begin{array}{llcllll}
\text{maximize} & x_{f'} & & & & & \\
\text{subject to} & & & & & & \\
 & x_{f'} & = & & - & & x_a \\
 & x_{B-f} & = & D_{B-f}x_N & + & \mathbf{1}^{(B-f)} & x_a \\
 & x_g & = & 1 & & & \\
 & x_j & \geq & 0 & & & \forall j \in E \cup \{a\} \setminus \{f, g\},
\end{array}
\tag{4.23}
$$

where $\mathbf{1}^{(B-f)}$ is the vector of all 1's indexed by $B - f$. Observe that the new objective variable is $-x_a$ whose maximization forces x_a to be zero if it is possible. It is easy to see that the original LP is feasible if and only if the optimum value is 0. This LP is not yet feasible, but a single pivot will make the dictionary primal feasible. More precisely, one can select any pivot (r, a) satisfying

$$r \in B - f \text{ and } d_{rg} = \min\{d_{ig} : i \in B - f\}.$$

Phase I of the simplex method is essentially to run Phase II with this auxiliary LP. The original objective function will be kept in the auxiliary dictionary throughout the Phase I. Since the objective function of the auxiliary LP is bounded by zero, Phase I must terminate in an optimal dictionary, say $\hat{D}$. If the optimal value is negative, the original LP is infeasible. Otherwise the optimal solution induces a feasible solution to the original LP. Moreover, one can find a feasible basis easily. There are two cases. The first case is when the auxiliary variable index a is nonbasic in the optimal basis. Then one can simply remove the a-column and the f'-row from $\hat{D}$ to obtain a feasible dictionary of the original LP. The second case is when the auxiliary variable index a is basic in $\hat{D}$. In this case $\hat{d}_{ag} = 0$ because the objective value ($= -x_a$) is zero at the basic solution. Now we make a pivot on (a, s) for any column s with $\hat{d}_{as} \neq 0$ and $s \in \hat{N} - g$. Such a pivot must exist, because otherwise x_a would be identically zero in the equality system of the auxiliary LP, that is impossible. Since $\hat{d}_{ag} = 0$, such a pivot will preserve the feasibility of the dictionary, and reduce this second case to the first case.

4.5 Implementing Pivot Operations

In this section, we discuss how to interpret pivoting in terms of certain matrix operations. The interpretation here is slightly different from one given in Section 4.3, and leads to an implementation that is convenient for sensitivity analysis to be given in Section 4.6. There are different ways for actual implementations but implementation details are beyond our scope.

Here we assume that an LP is given in standard form:

$$\begin{array}{llll} & \max & c^T x & \\ (4.24) & \text{subject to} & Ax & = b \\ & & x & \geq \mathbf{0}, \end{array}$$

where A is a given $m \times E$ matrix, $b \in R^m$ and $c \in R^E$. Since the row indices are not important, the rows are $\{1, 2, \ldots, m\}$ The only difference with LP in canonical form is the system $Ax = b$ is an equality system instead of inequality $Ax \leq b$. It is easy to reduce any LP in canonical form to an equivalent LP in standard form.

We assume that A is row full rank, and we are given a *basis* of A, that is, a subset $B \in E$ such that $|B| = m$ and the submatrix $A_{\cdot B}$ has rank m. We call both the index set B and the submatrix $A_{\cdot B}$ *basis* of A. We denote by $A_{\cdot B}^{-1}$ the *left inverse*, the unique $B \times m$ matrix T such that $T\, A_{\cdot B} = I^{(B)}$.

Using the (left) inverse $A_{\cdot B}^{-1}$, one easily convert an LP in standard form to one in dictionary form. First we solve $Ax = b$ in terms of x_B as

$$A_{\cdot B}^{-1}(A_{\cdot B} x_B + A_{\cdot N} x_N) = A_{\cdot B}^{-1} b$$

which gives

$$x_B = A_{\cdot B}^{-1} b - A_{\cdot B}^{-1} A_{\cdot N} x_N. \tag{4.25}$$

Now we substitute x_B in the objective function by

$$\begin{aligned} c^T x &= c_B^T x_B + c_N^T x_N \\ &= c_B^T \left(A_{\cdot B}^{-1} b - A_{\cdot B}^{-1} A_{\cdot N} x_N \right) + c_N^T x_N \\ &= c_B^T A_{\cdot B}^{-1} b + \left(c_N^T - c_B^T A_{\cdot B}^{-1} A_{\cdot N} \right) x_N. \end{aligned} \tag{4.26}$$

Let $\overline{B} = B + f$, $\overline{N} = N + g$ with new indices f and g, $\overline{E} = \overline{B} \cup \overline{N}$, and set $\overline{B} \times \overline{N}$ matrix D as

$$D = \begin{bmatrix} c_B^T A_{\cdot B}^{-1} b & c_N^T - c_B^T A_{\cdot B}^{-1} A_{\cdot N} \\ \\ A_{\cdot B}^{-1} b & -A_{\cdot B}^{-1} A_{\cdot N} \end{bmatrix}. \tag{4.27}$$

where the first row is indexed by f and the first column by g. Now we can rewrite an original LP in standard form to an equivalent LP in dictionary form

$$\begin{array}{lll} \text{maximize} & x_f & \\ \text{subject to} & & \\ & x_{\overline{B}} & = \; D x_{\overline{N}} \\ & x_g & = \; 1 \\ & x_j & \geq \; 0 \quad \forall j \in \overline{E} \setminus \{f, g\}. \end{array} \tag{4.28}$$

Hence we represent the dictionary associated with a basis B of the input matrix A in terms of the original matrix data A, b and c and the basis inverse $A_{\cdot B}^{-1}$. In other words, a dictionary can be computed as long as the input data and the current basis inverse $A_{\cdot B}^{-1}$ is at hand.

This leads to an implementation of any pivot algorithm based on **basis-inverse updates**. We might call this technique *revised pivot scheme*:

(R1) Compute the inverse $A_{\cdot B}^{-1}$ of an initial basis B.

(R2) Compute only the necessary part of the dictionary D using (4.27) for selecting a pivot (r, s).

(R3) Once a pivot (r, s) is chosen, update B by $B - r + s$ and compute the new inverse $A_{\cdot B}^{-1}$ from the old.

It should be remarked that the procedure (R3) is mathematically a simple matrix operation but its practical implementation is itself an important subject of matrix computation. Practicality means many different things, memory/time efficiency, accuracy, robustness, etc. We shall not discuss this issue here.

The simplex method with the revised pivot scheme is known as the *revised simplex method*. Most (if not all) commercial codes use some variations of the revised scheme. There are several reasons.

- Real-world LP problems have large and sparse input matrices. LP dictionaries, once pivots are performed, tend to become dense very quickly. Basis inverses are often much smaller than the input matrices and thus lots of memory can be saved.
- There are techniques, such as LU decomposition, of representing and updating basis inverses, to control the accuracy of computation.
- Selection of a pivot usually depends only on a fraction of the current dictionary entries. Thus updating the whole dictionary often wastes lots of time.

- Floating-point computation leads to accumulation of errors. It is extremely important to keep the original data and to recompute the basis inverse or whatever by reliable methods from time to time.

4.6 Computing Sensitivity

We consider LP in standard form in this section, because the necessary computation for the analysis to be discussed comes quite naturally with the revised pivot scheme in the previous section which is based on the standard form:

$$\begin{array}{lrl} \max & c^T x & \\ \text{subject to} & Ax & = b \\ & x & \geq \mathbf{0}, \end{array} \tag{4.29}$$

As we discussed in Section 4.5, a dictionary can be computed from the inverse $A_{\cdot B}^{-1}$ of the associated basis (of A) and the input matrix data as

$$D = \begin{bmatrix} c_B^T A_{\cdot B}^{-1} b & c_N^T - c_B^T A_{\cdot B}^{-1} A_{\cdot N} \\ \\ A_{\cdot B}^{-1} b & -A_{\cdot B}^{-1} A_{\cdot N} \end{bmatrix}. \tag{4.30}$$

The sensitivity analysis is to evaluate the stability of a given optimal solution or a given dual optimal solution. This question is usually modified to the same question for a given optimal basis, because this is much easier to compute. Also the stability of an optimal basis insures the stability of the associated (primal or dual) basic solutions.

The stability question for a given optimal basis (or dictionary) is to decide the widest ranges of (plus and minus) changes of a given input data for which its optimality is preserved. In principle any part of data is subject to change. However, certain analysis is much simpler to perform than others.

First of all, we remind that the optimality of a dictionary D means

- (Primal Feasibility) $A_{\cdot B}^{-1} b \geq \mathbf{0}$; and
- (Dual Feasibility) $c_N^T - c_B^T A_{\cdot B}^{-1} A_{\cdot N} \leq \mathbf{0}$.

Now, if one wants to decide the range of c_k for any fixed $k \in E$ for which the current dictionary stays optimal, it is just a matter of simplifying one variable inequality system $c_N^T - c_B^T A_{\cdot B}^{-1} A_{\cdot N} \leq \mathbf{0}$, where all c_j's except for c_k are constant.

Similarly, if one wants to decide the range of b_k for any fixed k for which the current dictionary stays optimal, it is a matter of simplifying one variable inequality system $A_{\cdot B}^{-1} b \geq \mathbf{0}$, where all b_i's except for b_k are constant.

These are two sensitivity analyses that are given by all standard LP software. It should be noted that the ranges one obtains by these analyses are the widest possible for the **optimality of the given basis**. It is possible that a wider range of c_k (b_k) preserves the **optimality** of the current primal (dual) **solution**[2]. In this sense, a more accurate sensitivity analysis is desirable, but to compute the widest range appears to be a much harder problem and requires more sophisticated techniques.

4.7 Dualization of Pivot Algorithms

As we presented in Proposition 4.6, a pivot operation on a position (r, s) in a dictionary D is the same thing as a pivot operation on (s, r) in the dual dictionary $-D^T$, if we interpret the result with its negative transpose.

Any pivot algorithm presented in terms of the primal dictionary, such as the simplex method or the criss-cross method, can be applied to the dual LP without looking at the dictionary of the dual LP. The *dual simplex method* is the simplex method applied to the dual LP described with the primal dictionary D. The *dual criss-cross* method is exactly the same method as the (primal) criss-cross method. Such a method is called *self-dual*.

Here is a description of the dual simplex method (Phase II), which needs an initial dual feasible basis D.

```
procedure DualSimplexPhaseII(D,g,f);
begin
    status:="unknown";
    while status="unknown" do
        Let R := {i ∈ B − f : d_ig < 0};
        if R = ∅ then
            status:="optimal";
        else
(L1)            select any r ∈ R;
            Let S := {j ∈ N − g : d_rj > 0};
            if S = ∅ then
                status:="dual_unbounded";
            else
(L2)                make a dual simplex pivot on (r, s) for any s ∈ S;
                replace D, B, N with the new D', B', N';
            endif;
        endif;
    endwhile;
    output (status, D); /* D is terminal */
end.
```

Above, a pivot on (r, s) is called a *dual simplex* pivot if

[2]This happens when the primal (dual) dictionary is degenerate.

(a) $d_{rs} > 0$;

(b) $-d_{fs}/d_{rs} = \min\{-d_{fj}/d_{rj} : j \in N - g \text{ and } d_{rj} > 0\}$.

One can easily show that by a dual simplex pivot the dual feasibility is preserved and the objective value d_{fg} of the primal basic solution does not increase.

Which simplex method should one use, primal or dual? This question is extremely important since they usually behave quite differently. We shall discuss this issue in Section 4.9.1.

4.8 Pivot Rules for the Simplex Method

The simplex method we presented in Section 4.4.3 admits multiple choices in selecting a pivot at each iteration. This flexibility is critical for making the simplex method behave favorably in one or more different criteria. For example, Bland's rule (Definition 4.14) was to make the simplex method finite. We shall present two simple rules that are useful for making the simplex method practically efficient.

Definition 4.16 *[Largest Coefficient Rule] In the step (L1) of SimplexPhaseII, select an index s from S having the largest cost d_{fs}.*

Definition 4.17 *[Steepest Descent Rule] In the step (L1) of SimplexPhaseII, select an index s from S having the largest δ_s, where*

$$\delta_s = \frac{d_{fs}}{\sqrt{\Sigma_{i \in B-f} d_{is}^2 + 1}}.$$

The quantity δ_s is essentially the amount of improvement in the objective function per unit length of the associated basic direction $x(B, s)$, since the denominator $\sqrt{\Sigma_{i \in B-f} d_{is}^2 + 1}$ is the length of the direction in the space of all variables except for x_f and x_g. The name "steepest descent" comes from the classical algorithm to find the minimum of a function, and employed here (and in standard textbooks) as it is for the maximization problem. Both rules rely on the intuition that when the associated measure d_{fs} or δ_s is the largest, the simplex pivot tends to make a relatively good improvement in the objective value.

It has been observed by extensive computational experiments that both rules usually reduce the number of simplex pivots considerably, comparing with, for example, Bland's rule or the random rule:

Definition 4.18 *[Random Rule (or Random-Edge Rule)] In the step (L1) of SimplexPhaseII, select an index s from S randomly.*

Furthermore, the steepest descent rule is considerably better than the largest coefficient rule in terms of the number of pivots for solving many benchmark LPs. The main problem to adopt the steepest descent rule is that the cost (time) to compute δ_s is quite expensive. Many efficient implementation uses certain approximation of δ_s that is much easier to compute.

It is important to note that there is no known *polynomial pivot rule*. That is, a rule that forces the simplex method to terminate in a number of pivots bounded above by a polynomial function of m and d, the number of basic and nonbasic variables, respectively. The positive resolution will be considered as a major breakthrough, because it would resolve several other important open problems in geometry, optimization and combinatorics, as well. There is a fascinating conjecture which had been open for over 35 years:

Conjecture 4.19 *[Liebling 1975] The expected number of pivots performed by Random Rule (Definition 4.18) is bounded above by a polynomial function of m and d.*

In 2011, Friedmann-Hansen-Zwick [15] found counterexamples to this conjecture, for which the Random Rule simplex method takes at least a superpolynomial number of pivots.

We also mention that there are important research developments on random pivot rules whose expected behavior can be analyzed. In particular, there are pivot rules due to Kalai-Kleitman [25] and to Sharir-Welzl [37] (1992) that attain sub-exponential bounds on the expected number of pivots. Since many of the practical pivot rules such as the largest coefficient rule were shown to require an exponential number of pivots in the worst case, any bounds such as sub-exponential bounds that do not grow as fast as any exponential function are interesting.

Finally, there is an indication that restricting to the simplex method might not be a good strategy to find a polynomial-time pivot algorithm. While the criss-cross method is exponential in general, the criss-cross method is conjectured to be an expected polynomial-time method if the variables are randomly permuted at the beginning of the course of the algorithm, see [13, 17].

Conjecture 4.20 *[Fukuda 2008] The expected number of pivots performed by the criss-cross method with random permutation of variables is bounded above by a polynomial function of m and d.*

For further readings on this subject, see Appendix A.

4.9 Geometry of Pivots

4.9.1 Geometric Observations of the Simplex Method

Until now, we have ignored the geometric aspects of linear programming. In this section, we present some geometric interpretations of pivot operations.

To get some feeling about the geometry of LPs, look at Ch. ETH example,

$$\begin{array}{lrcrcrcr}
\text{max} & 3x_1 & + & 4x_2 & + & 2x_3 & & \\
\text{subject to} & & & & & & & \\
\text{E1:} & 2x_1 & & & & & \leq & 4 \\
\text{E2:} & x_1 & & & + & 2x_3 & \leq & 8 \\
\text{E3:} & & & 3x_2 & + & x_3 & \leq & 6 \\
\text{E4:} & x_1 & \geq 0, & x_2 & \geq 0, & x_3 & \geq & 0
\end{array}$$

and depict the sequence of feasible solutions generated by the simplex method, see Figure 4.7.

	g	1	2	3
f	0	3	4	2
4	4	−2	0	0
5	8	−1	0	−2
6	6	0	−3	−1

⇓ Pivot 1

	g	1	2	5
f	8	2	4	−1
4	4	−2	0	0
3	4	−1/2	0	−1/2
6	2	1/2	−3	1/2

⇓ Pivot 2

	g	1	6	5
f	32/3	8/3	−4/3	−1/3
4	4	−2	0	0
3	4	−1/2	0	−1/2
2	2/3	1/6	−1/3	1/6

⇓ Pivot 3

	g	4	6	5
f	16	−4/3	−4/3	−1/3
1	2	−1/2	0	0
3	3	1/4	0	−1/2
2	1	−1/12	−1/3	1/6

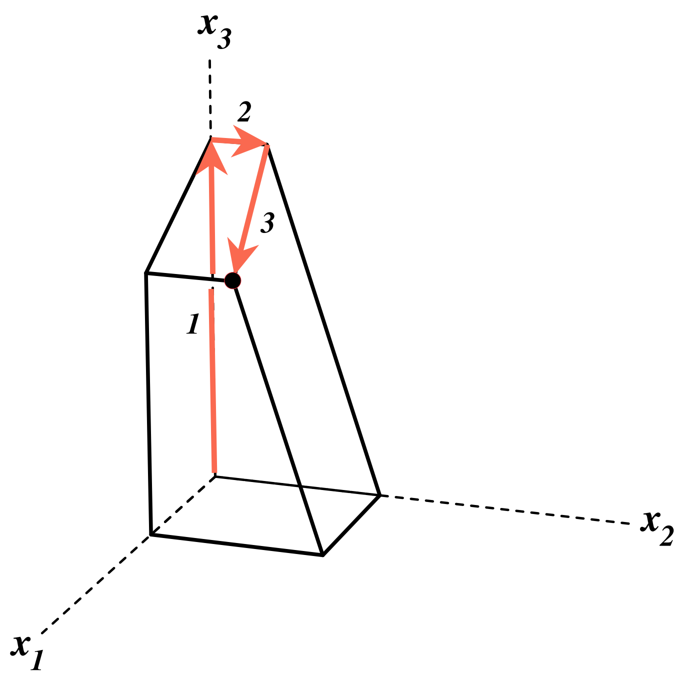


Figure 4.7: Feasible Region Ω and the Trace of the Simplex Method

The simplex method starts with the initial basic solution $(0,0,0)$, and traces the path:

$$(0,0,0) \longrightarrow (0,0,4) \longrightarrow (0,2/3,4) \longrightarrow (2,3,1).$$

As we see in Figure 4.7, the simplex traces the boundary edges of the feasible region by moving from a vertex (i.e., an extreme point in geometric language) to a neighboring vertex which attains a better objective value.

We will prove formally in Section 4.9.3 that every basic feasible solution is in fact an extreme point of the feasible region. (The definition of extreme point will be given later.)

From this geometric interpretation of the simplex method, one sees how the simplex method will be affected by the combinatorial structure of the feasible region. For example, the **number of extreme points** of a feasible region can be a very important measure for the worst case bound for the number of simplex pivots. The combinatorial diameter of a convex polyhedron determines how efficient an ideal simplex method can be. The **combinatorial diameter** of Ω is the maximum of the lengths (in terms of number of edges) of shortest paths between any pair of vertices. There is a famous conjecture[3] by Hirsch that the diameter is at most $n - d$ for any d-dimensional convex polytope (i.e. **bounded** convex polyhedron) given by a system of n inequalities. In the example above, $d = 3$ and $n = 6$, the conjecture says one can connect any vertex to another by a path with at most 3 edges. The conjecture is true for $d = 3$ and for various special cases, yet it is open in general.

Another interesting question is whether one should solve the primal LP or the dual LP, for any given LP. If the size of matrix A in a given LP in canonical form is $m \times d$, the feasible region Ω in the primal LP is in R^d, and the dual Ω^* in R^m. The number n of inequalities is $m + d$ for both problems. It has been observed (and proved for special cases) that the Hirsch bound is a good measure for comparing the actual diameter. This means the Hirsch bound m for the primal and d for the dual often indicate which polyhedron has smaller diameter. In short, the primal region tends to have smaller diameter if its dimension d is larger than the dual dimension, and larger diameter otherwise. Thus try to solve whichever problem whose dimension is larger. If they are roughly equal, maybe it does not matter which one.

In Figure 4.8, we present small examples of Ch. ETH type where d is fixed to 3 and m is chosen to be $3, 10, 50, 100$. The values $\mathrm{v}(\Omega)$ and $\mathrm{diam}(\Omega)$ represent the number of vertices and the diameter of a convex polyhedron Ω.

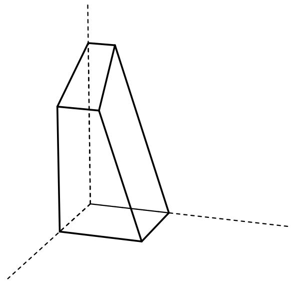
First of all, it is clear that the boundary of the primal feasible region becomes more complex as m grows. The dual feasible region is in m-dimensional space and thus is hard to draw, but the same remark is true for the dual regions. There are interesting contrasts between the primal and the dual feasible regions: the primal region has much fewer vertices than the dual region, but it has considerably larger diameter than the dual region. This trend becomes stronger as m grows. In fact, when m is much larger than d, it is almost always faster to solve the LP by the dual simplex method, the simplex method applied to the dual problem, than by the primal simplex method. One can intuitively justify this observation by the difference in diameters. In the dual LP, there is always a relatively short route to the optimal vertex. Another way to understand this phenomenon is by looking at certain special LP models and/or special (randomized) types of the simplex method.

[3]The conjecture was disproved by Santos [34] in 2010. Yet, the Hirsch bound is only slightly (by a small constant) exceeded by the construction and thus a slightly modified bound might be still valid.

Finally we see that the Hirsch bound is not valid for unbounded polyhedra. In the case $m = 10$, the dual polyhedron has diameter 4, which is larger than the Hirsch bound 3. This fact was proved first by Klee and Walkup in 1967.

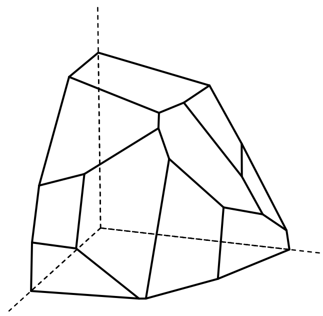
$d = 3$, $m = 3$

$\mathrm{v}(\Omega) =$	8	$\mathrm{diam}(\Omega) =$	3
			$=$
$\mathrm{v}(\Omega^*) =$	6	$\mathrm{diam}(\Omega^*) =$	3



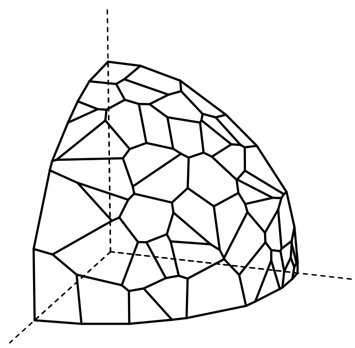
$d = 3$, $m = 10$

$\mathrm{v}(\Omega) =$	22	$\mathrm{diam}(\Omega) =$	6
	$\wedge$		$\vee$
$\mathrm{v}(\Omega^*) =$	92	$\mathrm{diam}(\Omega^*) =$	4



$d = 3$, $m = 50$

$\mathrm{v}(\Omega) =$	102	$\mathrm{diam}(\Omega) =$	12
	$\wedge$		$\vee$
$\mathrm{v}(\Omega^*) =$	5983	$\mathrm{diam}(\Omega^*) =$	4



$d = 3$, $m = 100$

$\mathrm{v}(\Omega) =$	202	$\mathrm{diam}(\Omega) =$	16
	$\wedge$		$\vee$
$\mathrm{v}(\Omega^*) =$	44301	$\mathrm{diam}(\Omega^*) =$	?

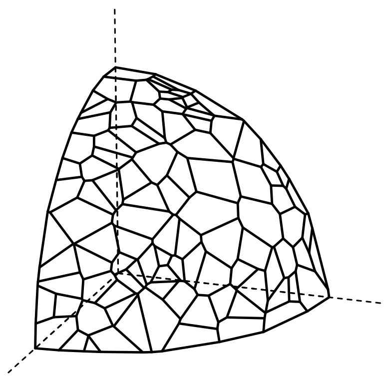


Figure 4.8: Primal Feasible Regions for Increasing n in R^3

4.9.2 Three Paradigms of LP Algorithms

We learned two pivot algorithms in this chapter, the simplex method and the criss-cross method. There are many different variations of these methods as well. Later we present yet another algorithm paradigm to linear programming, the interior-point algorithm.

The simplex method follows an edge path on the feasible region, the criss-cross method follows a path in the whole space but stays on the lines generated by the constraint halfspaces (1-skeleton of the arrangement of hyperplanes). The interior-point method follows a well-defined nonlinear path in the whole space, see Figure 4.9. The criss-cross method admits the simplest correct implementation. In terms of practicality the other two methods win by far, at least with our present knowledge. And only the interior-point method enjoys polynomial time complexity (for the moment). One might find these different characteristics of LP algorithms rather annoying, but I consider them as clear indications for the richness of LP Theory with fascinating mysteries.

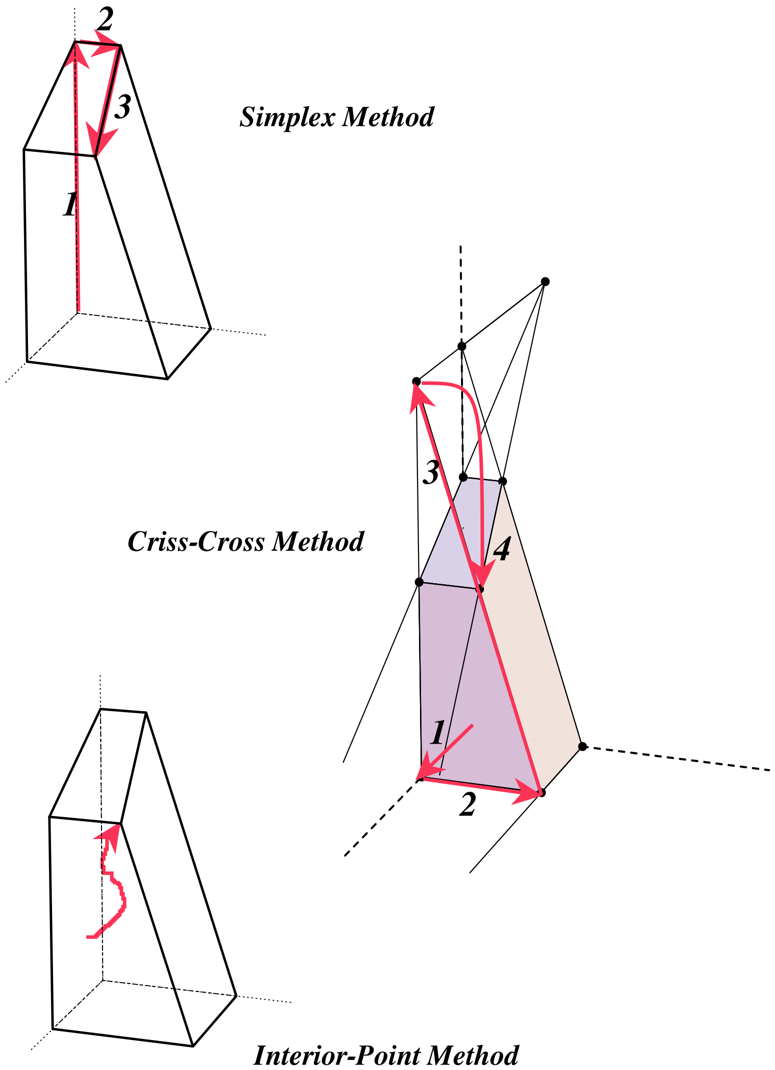


Figure 4.9: Three Paradigms of LP Algorithms

4.9.3 Formal Discussions

Consider an LP in dictionary form,

$$\text{(4.31)} \qquad \begin{array}{lllll} \text{maximize} & x_f & & & \\ \text{subject to} & & & & \\ & x_{B_0} & = & Dx_{N_0} & \\ & x_g & = & 1 & \\ & x_j & \geq & 0 & \forall j \in E \setminus \{f, g\}. \end{array}$$

We denote by Ω the set of all feasible solutions:

$$\text{(4.32)} \qquad \Omega = \{x \in R^E : x_{B_0} = Dx_{N_0}, x_g = 1 \text{ and } x_j \geq 0 \quad \forall j \in E \setminus \{f, g\}\}.$$

where we set $E = B_0 \cup N_0$. For each subset S of E, set

$$\text{(4.33)} \qquad \Omega(S) = \{x_S \in R^S : x \in \Omega\},$$

which is the orthogonal projection of Ω onto the subspace space R^S. Two sets $P \in R^E$ and $Q \in R^S$ are called *affinely equivalent* (denoted by $P \sim Q$) if P and Q can be mapped to each other by some affine bijections. Then one can verify the following:

$$\text{(4.34)} \qquad \Omega \sim \Omega(N - g) \quad \text{for any nonbasis } N \text{ of the dictionary system.}$$

The less trivial part of constructing an affine map from $\Omega(N - g)$ can be done by using the basic fact that $x_g = 1$ and all basic variables x_B are uniquely determined by the nonbasic variables x_N.

It is easy to see that the feasible region and its projections can be represented as

$$\text{(4.35)} \qquad \Omega(S) = \{x \in R^S : Ax \leq b\}.$$

for some $m \times S$ matrix A and an S-vector b. Such a representation is useful to prove certain properties of $\Omega(S)$.

In general, the set of solutions to a (finite) system of linear inequalities in R^S is called a *convex polyhedron*. The feasible region to a linear inequality system is exactly such a set. For dimension $d = |S|$ two or three, convex polyhedra can be visualized, but higher dimensional polyhedra are not easy to "see".[4] In order to understand their structures, it is convenient to make use of different representations of convex polyhedra. Here we rely only on the definition itself, the inequality representation (or *H-representation* where H refers to halfspaces).

For example, for Chateau ETH Problem, Example 1.1:

$$\begin{array}{lrcrcrcl} \text{max} & 3x_1 & + & 4x_2 & + & 2x_3 & & \\ \text{subject to} & & & & & & & \\ \text{E1:} & 2x_1 & & & & & \leq & 4 \\ \text{E2:} & x_1 & & & + & 2x_3 & \leq & 8 \\ \text{E3:} & & & 3x_2 & + & x_3 & \leq & 6 \\ \text{E4:} & x_1 & \geq 0, & x_2 & \geq 0, & x_3 & \geq & 0 \end{array}$$

[4]There are some techniques, such as Schlegal Diagrams and Gale Transformations, to visualize polyhedra with special configurations.

the associated canonical system is

$$\begin{array}{lrcrcrcrcr} \text{maximize} & x_f & = & 0x_g & + & 3x_1 & + & 4x_2 & + & 2x_3 \\ \text{subject to} & x_4 & = & 4x_g & - & 2x_1 & & & & \\ & x_5 & = & 8x_g & - & x_1 & & & - & 2x_3 \\ & x_6 & = & 6x_g & & & - & 3x_2 & - & x_3 \\ & & & \multicolumn{7}{r}{x_g = 1,\ x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0} \end{array}$$

and the feasible region is

$$\begin{aligned} \Omega = \{x = (x_1, x_2, \ldots, x_6, x_f, x_g)^T :& \\ & x_f = 3x_1 + 4x_2 + 2x_3 \\ & x_4 = 4x_g - 2x_1 \\ & x_5 = 8x_g - x_1 - 2x_3 \\ & x_6 = 6x_g - 3x_2 - x_3 \\ & x_g = 1 \text{ and} \\ & x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0\}. \end{aligned} \tag{4.36}$$

Setting $S_1 = \{1, 2, 3\}$, we have

$$\begin{aligned} \Omega(\{1,2,3\}) = \{x \in R^{S_1} :& 2x_1 \leq 4, x_1 + 2x_3 \leq 8, 3x_2 + x_3 \leq 6, \\ & x_1 \geq 0, x_2 \geq 0, x_3 \geq 0\} \end{aligned} \tag{4.37}$$

Figure 4.10 shows this region which is affinely equivalent to Ω.

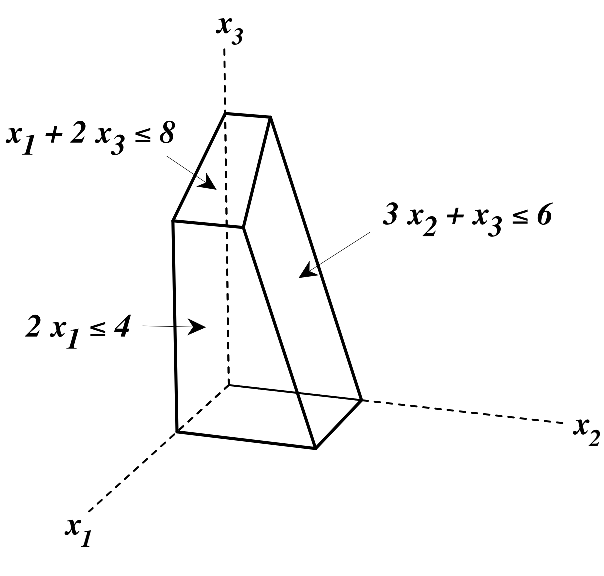


Figure 4.10: Feasible Region $\Omega(\{1, 2, 3\})$

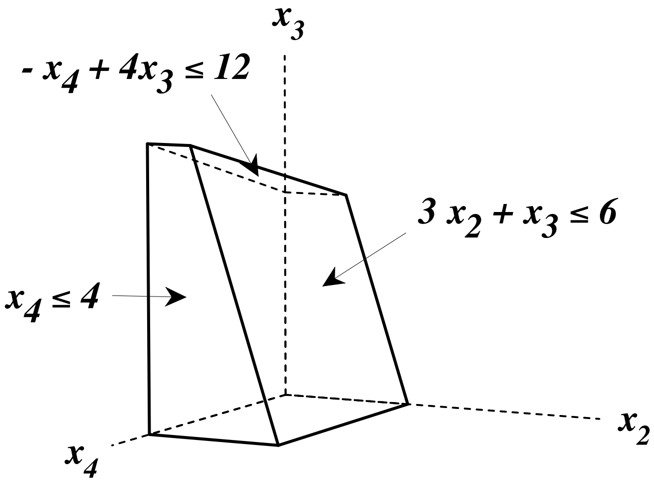


Figure 4.11: Feasible Region $\Omega(\{4,2,3\})$

On the other hand, by setting $S_2 = \{4,2,3\}$, we have

$$\Omega(\{4,2,3\}) = \{x \in R^{S_2} : x_4 \le 4, -x_4 + 4x_3 \le 12, 3x_2 + x_3 \le 6, \quad x_4 \ge 0, x_2 \ge 0, x_3 \ge 0\} \tag{4.38}$$

This can be verified if one performs a pivot on $(4,1)$ (exchanging x_4 and x_1) in the initial dictionary to get

$$\begin{array}{lrcrcrcrcr}
\text{maximize} & x_f & = & 6x_g & - & 3/2x_4 & + & 4x_2 & + & 2x_3 \\
\text{subject to} & x_1 & = & 2x_g & - & 1/2x_4 & & & & \\
 & x_5 & = & 6x_g & + & 1/2x_4 & & & - & 2x_3 \\
 & x_6 & = & 6x_g & & & - & 3x_2 & - & x_3 \\
 & & & \multicolumn{7}{r}{x_g = 1,\ x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \ge 0.}
\end{array}$$

Figure 4.11 depicts the region $\Omega(\{4,2,3\})$. The regions $\Omega(\{1,2,3\})$ and $\Omega(\{4,2,3\})$ are affinely equivalent. They simply use different coordinate systems to represent the same object Ω. Note that the origins are the basic solutions of the associated bases. One might already foresee that the basic solutions are "extreme" points of the feasible region.

Let us formally define a few geometric notions including "extreme points" that are useful when we talk about the geometry of feasible regions and basic solutions.

Let x^1, x^2 be points in R^d, and let S be a subset of R^d. A *convex combination* of x^1 and x^2 is a point $\lambda x^1 + (1-\lambda)x^2$ for some $0 \le \lambda \le 1$. The line segment between x^1 and x^2 is the set of all convex combinations of them:

$$[x^1, x^2] = \{x \in R^d : x = \lambda x^1 + (1-\lambda)x^2, 0 \le \lambda \le 1\}. \tag{4.39}$$

A subset S of R^d is called *convex* if the line segment between any two points in S is contained in itself. See Figure 4.12. A point x is called an *extreme point* of S if $x \in S$ and x cannot be represented as a convex combination of two points of S different from x.

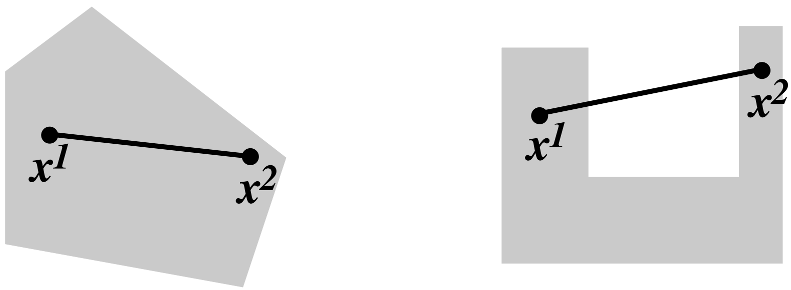


Figure 4.12: Convex and Nonconvex Sets

Proposition 4.21 *The feasible region Ω is convex.*

Proof. We use the representation $\Omega = \{x \in R^E : Ax \le b\}$. Let x^1, x^2 be points of Ω, and let x be a convex combination of x^1 and x^2. Since

$$\begin{aligned} Ax &= A(\lambda x^1 + (1-\lambda)x^2) = \lambda A x^1 + (1-\lambda)Ax^2 \\ &\le \lambda b + (1-\lambda) b = b, \end{aligned}$$

the set Ω is convex. ∎

A solution $\overline{x}$ to a system of inequalities $Ax \le b$ is said to be *elementary* if it is the unique solution to $A^1 x = b^1$, where $A^1 x \le b^1$ is the maximal subsystem of $Ax \le b$ satisfied by $\overline{x}$ with equality, i.e. determined uniquely by

$$A = \begin{bmatrix} A^1 \\ A^2 \end{bmatrix}, b = \begin{bmatrix} b^1 \\ b^2 \end{bmatrix}, A^1\overline{x} = b^1, A^2\overline{x} < b^2.$$

Theorem 4.22 *Let $\Omega = \{x \in R^E : Ax \le b\}$. Then a point $x \in \Omega$ is an extreme point if and only if it is an elementary solution to $Ax \le b$.*

Proof. Let $\overline{x}$ be an elementary solution. Suppose $\overline{x}$ be a convex combination of two points x^1 and x^2 in Ω, i.e. $\overline{x} = \lambda x^1 + (1-\lambda)x^2$ for some λ with $0 \le \lambda \le 1$. Now pick up any inequality $A_i x \le b_i$ from $A^1 x = b^1$. We claim that both x^1 and x^2 must satisfy this inequality with equality, since otherwise

$$\begin{aligned} A_i\overline{x} &= A_i(\lambda x^1 + (1-\lambda)x^2) = \lambda A_i x^1 + (1-\lambda)A_i x^2 \\ &< \lambda b_i + (1-\lambda) b_i = b_i, \end{aligned}$$

a contradiction to the choice of i. This means both x^1 and x^2 satisfy $A^1 x = b^1$. Since $\overline{x}$ is elementary, all $\overline{x}$, x^1 and x^2 must be equal. This implies $\overline{x}$ is an extreme point.

The other direction is left for exercise. ∎

Corollary 4.23 *The notion of elementary solutions depends only on the set of solutions to the given system.*

Theorem 4.24 *Every basic feasible solution to an LP in dictionary form is an extreme point of its feasible region.*

Proof. Consider an LP in dictionary form. Represent the constraints

$$x_B = Dx_N, x_{E\setminus\{f,g\}} \geq \mathbf{0}, x_g = 1$$

as $Ax \leq b$ by using two opposite inequalities for each equality. Thus the feasible region is $\{x : Ax \leq b\}$.

Note that the basic solution $x(B, g)$ is uniquely determined by $x_B = Dx_N$, $x_g = 1$ and $x_{N-g} = \mathbf{0}$. Clearly these equalities are contained in the maximal equality subsystem w.r.t. $x(B, g)$ of $Ax \leq b$, and thus the subsystem must have a unique solution. This implies $x(B, g)$ is an elementary solution to $Ax \leq b$. By Theorem 4.22, $x(B, g)$ is an extreme point of the feasible region. ∎

4.10 Examples of Pivot Sequences

4.10.1 An example of cycling by the simplex method

$$
\begin{array}{llcrcrcrcr}
\text{maximize} & x_f & = & 0 & + & x_1 & - & 2x_2 & + & x_3 \\
\text{subject to} & x_4 & = & 0 & - & 2x_1 & + & x_2 & - & x_3 \\
 & x_5 & = & 0 & - & 3x_1 & - & x_2 & - & x_3 \\
 & x_6 & = & 0 & + & 5x_1 & - & 3x_2 & + & 2x_3 \\
 & & & & & & & & & x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0
\end{array}
$$

	g	1	2	3
f	0	1	−2	1
4	0	−2	1	−1
5	0	**−3**	−1	−1
6	0	5	−3	2

⟸

	g	1	6	3
f	0	−7/3	2/3	−1/3
4	0	−1/3	−1/3	−1/3
5	0	−14/3	1/3	−5/3
2	0	5/3	**−1/3**	2/3

⇓ ⇑

	g	5	2	3
f	0	−1/3	−7/3	2/3
4	0	2/3	5/3	**−1/3**
1	0	−1/3	−1/3	−1/3
6	0	−5/3	−14/3	1/3

	g	1	6	4
f	0	−2	1	1
3	0	−1	−1	**−3**
5	0	−3	2	5
2	0	1	−1	−2

⇓ ⇑

	g	5	2	4
f	0	1	1	−2
3	0	2	5	−3
1	0	−1	−2	1
6	0	−1	**−3**	−1

⟹

	g	5	6	4
f	0	2/3	−1/3	−7/3
3	0	1/3	−5/3	−14/3
1	0	**−1/3**	2/3	5/3
2	0	−1/3	−1/3	−1/3

Note: How can one find an example of cycling? By chance? Not quite. The key idea lies in a geometric interpretation of the dual simplex method in the space of the primal problem.

4.10.2 Simplex Method (Phase II) applied to Chateau ETH problem

$$\begin{array}{llcrcrcrcr} \text{maximize} & x_f & = & 0 & + & 3x_1 & + & 4x_2 & + & 2x_3 \\ \text{subject to} & x_4 & = & 4 & - & 2x_1 & & & & \\ & x_5 & = & 8 & - & x_1 & & & - & 2x_3 \\ & x_6 & = & 6 & & & - & 3x_2 & - & x_3 \\ & & & & & \multicolumn{5}{r}{x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0} \end{array}$$

	g	1	2	3
f	0	3	4	2
4	4	−2	0	0
5	8	−1	0	−2
6	6	0	−3	−1

$\Downarrow$

	g	1	2	5
f	8	2	4	−1
4	4	−2	0	0
3	4	−1/2	0	−1/2
6	2	1/2	−3	1/2

$\Downarrow$

	g	1	6	5
f	32/3	8/3	−4/3	−1/3
4	4	−2	0	0
3	4	−1/2	0	−1/2
2	2/3	1/6	−1/3	1/6

$\Downarrow$

	g	4	6	5
f	16	−4/3	−4/3	−1/3
1	2	−1/2	0	0
3	3	1/4	0	−1/2
2	1	−1/12	−1/3	1/6

4.10.3 Criss-Cross method applied to Chateau ETH problem

$$\begin{array}{llcl}
\text{maximize} & x_f & = & 0 + 3x_1 + 4x_2 + 2x_3 \\
\text{subject to} & x_4 & = & 4 - 2x_1 \\
& x_5 & = & 8 - x_1 - 2x_3 \\
& x_6 & = & 6 - 3x_2 - x_3 \\
& & & x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0
\end{array}$$

	g	1	2	3
f	0	3	4	2
4	4	−2	0	0
5	8	−1	0	−2
6	6	0	−3	−1

⇓

	g	4	2	3
f	6	−3/2	4	2
1	2	−1/2	0	0
5	6	1/2	0	−2
6	6	0	−3	−1

⇓

	g	4	6	3
f	14	−3/2	−4/3	2/3
1	2	−1/2	0	0
5	6	1/2	0	−2
2	2	0	−1/3	−1/3

⇓

	g	4	6	2
f	18	−3/2	−2	−2
1	2	−1/2	0	0
5	−6	1/2	2	6
3	6	0	−1	−3

⇓

	g	4	6	5
f	16	−4/3	−4/3	−1/3
1	2	−1/2	0	0
2	1	−1/12	−1/3	1/6
3	3	1/4	0	−1/2

4.10.4 Criss-Cross method applied to a cycling example

$$\begin{array}{lrcrcrcrcr} \text{maximize} & x_f & = & 0 & + & x_1 & - & 2x_2 & + & x_3 \\ \text{subject to} & x_4 & = & 0 & - & 2x_1 & + & x_2 & - & x_3 \\ & x_5 & = & 0 & - & 3x_1 & - & x_2 & - & x_3 \\ & x_6 & = & 0 & + & 5x_1 & - & 3x_2 & + & 2x_3 \\ & & & & & \multicolumn{5}{r}{x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0} \end{array}$$

	g	1	2	3
f	0	1	−2	1
4	0	−2	1	−1
5	0	−3	−1	−1
6	0	5	−3	2

⇓

	g	4	2	3
f	0	−1/2	−3/2	1/2
1	0	−1/2	1/2	−1/2
5	0	3/2	−5/2	1/2
6	0	−5/2	−1/2	−1/2

⇓

	g	4	2	1
f	0	−1	−1	−1
3	0	−1	1	−2
5	0	1	−2	−1
6	0	−2	−1	1

4.10.5 Simplex method with Bland's rule applied to a cycling example

$$\begin{array}{lrcrcrcrcr} \text{maximize} & x_f & = & 0 & + & x_1 & - & 2x_2 & + & x_3 \\ \text{subject to} & x_4 & = & 0 & - & 2x_1 & + & x_2 & - & x_3 \\ & x_5 & = & 0 & - & 3x_1 & - & x_2 & - & x_3 \\ & x_6 & = & 0 & + & 5x_1 & - & 3x_2 & + & 2x_3 \\ & & & & & \multicolumn{5}{r}{x_1,\ x_2,\ x_3,\ x_4,\ x_5,\ x_6 \geq 0} \end{array}$$

	g	1	2	3
f	0	1	−2	1
4	0	−2	1	−1
5	0	−3	−1	−1
6	0	5	−3	2

⇓

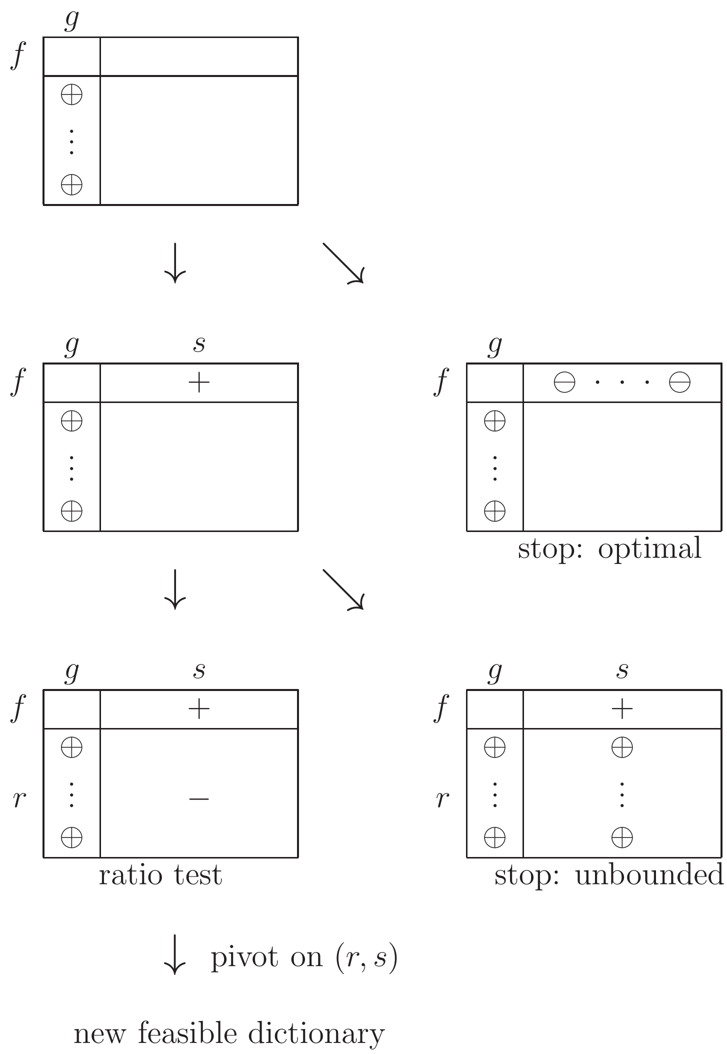
	g	4	2	3
f	0	−1/2	−3/2	1/2
1	0	−1/2	1/2	−1/2
5	0	3/2	−5/2	1/2
6	0	−5/2	−1/2	−1/2

⇓

	g	4	2	1
f	0	−1	−1	−1
3	0	−1	1	−2
5	0	1	−2	−1
6	0	−2	−1	1

4.11 Visual Description of Pivot Algorithms

4.11.1 Simplex Method



4.11.2 Criss-Cross Method