SOME CONTRIBUTIONS TO THE THEORY OF MATHEMATICAL PROGRAMMING

Chandra P. Saksena

A Thesis Submitted for the Degree of PhD at the University of St Andrews

1970

Full metadata for this item is available in St Andrews Research Repository at:
http://research-repository.st-andrews.ac.uk/

Please use this identifier to cite or link to this item:
http://hdl.handle.net/10023/13734

This item is protected by original copyright
THESIS for Ph.D.

MAY 1970
I hereby declare that this thesis has been composed by myself; that the work of which it is a record has been done by myself; and, that it has not been accepted in any previous application for any higher degree. This research concerning the Theory of Mathematical Programming was undertaken on 16th October 1967, the date of my admission as a research student under Ordinance General No. 12 for the degree of Doctor of Philosophy (Ph.D.)

5th May 1970.

Chandra P. Saksena
I hereby declare that the conditions of the Ordinance and Regulations for the degree of Doctor of Philosophy (Ph.D.) at the University of St. Andrews have been fulfilled by the candidate, Chandra P. Saksena.

Professor A.J. Cole
SOME CONTRIBUTIONS
TO THE THEORY OF
MATHEMATICAL PROGRAMMING
To my parents

and

Maya P. Saksena
Acknowledgments

The work contributed in this thesis on the Theory of Mathematical Programming could be brought to fruition mainly because of the scholarship offered by the University of St. Andrews since it enabled me to come from my distant country (India) to work at St. Andrews in October 1967. I acknowledge with gratitude the financial help thus offered to me by the University Authorities and in particular I am most grateful to Professor A.J. Cole for his recommendation to this effect.

I am grateful and indebted to Professor A.J. Cole also for his valuable guidance and the time and energy that he devoted in supervising my work. I am thankful to him also for the strenuous task of going through the manuscript of this thesis. I am also grateful to Professor O.J. Dahl of the University of Oslo with whom I had some helpful discussions at an early stage in this project.

I would also like to express thanks to my colleagues in the Computing Laboratory, St. Andrews University, for their help and cooperation throughout this work.

My appreciation and thanks are due to Miss Susan M. Watt for her hard work and the excellent typing of the manuscript. I should also like to thank Mrs Cecile Abramson for her valuable help in typing a part of the thesis.

Finally, I am most indebted to my wife for her help in checking the thesis against the manuscript and her cheerful acceptance of a part-time husband during the long period that I remained engaged on this research project.
CONTENTS

CHAPTER 1. BACKGROUND, HISTORY AND GENERAL INTRODUCTION 1
Background and history 1
Optimisation problems 1
Programming Problems 2
History 3
Computer Codes 7
Introduction 7

CHAPTER 2. THE GENERAL LINEAR PROGRAMMING PROBLEM 10
Formulation of the general linear programming problem 10
Formulation 1 10
Formulation 2 11
The Multiplex Method of Linear Programming 12
The Multiplex flow diagram and the algorithm 19
The flow diagram 19
The algorithm 20

CHAPTER 3. THE BOUNDING HYPERPLANE METHOD OF LINEAR PROGRAMMING - PART I 22
The Inequalities 22
Decision rules 27
The pivotal transformations 31
The unbounded and non-feasible problems 31
CHAPTER 6.

(Contd.)

The extended algorithm 86
Phase 0 86
New tableau and the evaluations 86
Phases 1 and 2 88
Degeneracy 88
Illustrations 89

CHAPTER 7. COMPARATIVE STUDY OF THE SIMPLEX AND B.H.P. METHODS

Number of iterations required by the B.H.P.M. vis-a-vis Simplex 99
Time taken by the two methods 99

CHAPTER 8. SUGGESTIONS FOR FURTHER RESEARCH

Extensions to quadratic programming 102
Accuracy in linear programming 107
Modifications to reinversion process 107
Modifications to the equality solution techniques 108

BIBLIOGRAPHY

111
Chapter 1

BACKGROUND, HISTORY AND GENERAL INTRODUCTION

1.1 Background and history.

1.1.1 Optimisation problems. The class of problems that seek to optimise a function of a number of variables (functions) subject to certain constraints on the variables (functions) are, in general, called optimisation problems. These have long interested and intrigued mathematicians; Euclid in the 3rd Century B.C., for example, mentioned a number of optimisation problems in his Elements. Most of the optimisation problems, however, defied rigorous solution until the development of such mathematical disciplines as the calculus of variations, differential calculus and the like. These have subsequently been applied, with considerable success, to the solution of a wide range of classical optimisation problems arising mainly in the fields of engineering sciences, geometry and physical sciences. In the past two decades or so a new class of optimisation problems, termed "programming" problems, have been encountered – arising mainly in the fields of economic theory, business, military and industrial operations. These are usually not emenable to solution by the classical method of calculus.
Classical methods using lagrangian multiplier techniques \(^1\) have however been developed (Klien 1955\(^{[50]}\)) for resolving the non-negativity requirements of the variables in the problems with inequality constraints. These are found to be most impractical except for the solution of 'toy' problems (Charnes and Cooper, 1955\(^{[11]}\)).

1.1.2 Programming Problems: The general programming problem is concerned with maximising or minimising a given objective function subject to a set of restrictions; mathematically, it may be stated as under:

Find a vector \( X = (X_1, X_2, \ldots, X_n) \) which maximises (or minimises)

\[
F = f(X),
\]

subject to

\[
\begin{align*}
g_i(X) & \geq 0, \quad i = 1, 2, \ldots, m_v, \\
h_k(X) & = 0, \quad k = 1, 2, \ldots, m_r, \\
X & \geq 0.
\end{align*}
\]

If the functions \( f \), \( \{g_i\} \) and \( \{h_k\} \) are all linear the problem (1.1.1) is called the linear programming problem. If any of these functions is non-linear, it is called the non-linear programming problem; in particular, if \( f \) is a quadratic function and \( \{g_i\} \) and \( \{h_k\} \) are linear, then it is a quadratic programming problem.

1. Developments to Lagrangian multiplier techniques have, however, been continued to be made to tackle both the linear and non-linear programming problems (Everett 1963\(^{[20]}\), Falk, 1967\(^{[25]}\)).

2. The term 'linear programming' was suggested by T.C. Koopmans to G.B. Dantzig in 1951 as an alternative to the earlier form 'programming in a linear structure' (Dantzig 1948\(^{[16]}\)).
The linear programming problem may, further, be classified as the general structure or special structure programming problem depending upon whether the coefficient matrices in $\{q_k\}$ and $\{h_k\}$ have the former or the latter structure. This thesis is concerned with the general linear programming structure and quadratic programming problems.

1.1.3 History. The famous mathematician J.B.J. Fourier (1826 [28]) appears to be the first to have come across a linear programming problem while attempting to find the least maximum deviation to a system of linear equations. Interestingly his suggestion for the solution of the above problem by vertex to vertex descent to the minimum is also the principle behind the Simplex Method of linear programming developed by G.B. Dantzig in 1947. Later the well known mathematician M.Ch.J. de la Vallee Poussin (1911[64]) also, while considering the above problem, suggests a similar solution.

Subsequently, the Russian mathematician L.V. Kantrovich (1939, 1942[74,75]), while considering the application of mathematics to production problems and in particular to transportation problems, seems to be the first to have recognised the well defined mathematical structure of production problems that were amenable to numerical solution. His joint paper with

3. The case when the function $f$ has a positive semidefinite quadratic form.

4. Fourier reduced this problem to finding the lowest point of a polyhyderal set.
M.K. Gavurin (1919) describes in detail the theory of the transshipment problem. The computational algorithm given in this paper is, however, incomplete. During this period Hitchcock (1941) independently formulated and solved the transportation problem and later G. Stigler (1945) described the famous 'diet problem'.

However, despite the recognition of linear programming problems in one form or the other, it was only in 1947 that the general linear programming problem was formulated in precise mathematical terms by G.B. Dantzig and others in the U.S. department of the Air Force which then constituted a group called project SCOOP (scientific computation of optimal programs). The most outstanding contribution of project SCOOP was the Simplex technique for the solution of the general linear programming problem. It was presented by G.B. Dantzig (1949) at the historic conference (June 20 to 24, 1949) held in Chicago by the Cowles Commission for Research in Economics and whose proceedings were brought out under the direction of T.C. Koopmans in 1951. The important results on duality based on unpublished notes of J. Von Neuman were also, for the first time, presented by Dantzig (1951) and Gale, Kuhn and Tucker (1951) at the same conference.

5. T.C. Koopmans (1947) too independently later solved this problem.

6. The diet problem was, in fact, formulated and approximately proposed in 1941 by Jerome Cornfield in an unpublished memorandum and treated by George Stigler by what he calls an 'experimental procedure'.
Subsequently an important and, in fact, the first symposium in Linear Inequalities and Programming was held (June 14 to 16, 1951) in Washington under the joint auspices of the Air Force and the National Bureau of Standards. The proceedings of the symposium were published in 1952, under the direction of A. Orden and L. Goldstein. The proceedings of these symposia are of significance in the history of linear programming as they constituted for a number of years the only general source of information and no doubt provided great stimulus to a number of individuals and research organisations for further research, developments and extensions in the field of linear programming.

Subsequently a number of symposia in linear programming (or covering the linear programming field) have been held and also a number of journals from different computing, operational research and allied fields sprang up all over the world. This has mainly been due to remarkable growth in the applications of linear programming to industrial problems and the simultaneous development of new techniques for the solution of linear programming problems. Of the latter, the Dual Simplex Method developed by Lemke in 1954, may particularly be mentioned being concerned with those aspects of the general linear programming problem that, though often peculiar to the former, were not covered by the Simplex Method. As a result the Composite Simplex-Dual Simplex algorithm appeared in 1954. There are a number of other methods which can be used for solving the
linear programming problem. These include; Relaxation Method, Motzkin and Schoenberg [58]; Projection method, Tompkins [75]; the Double Gradient and Multiplex Methods, Frisch[31,30]; the Hungarian Method, Kuhn [53]; the Stepping Stone Method, Charnes and Cooper [10] and a refinement to this method called the MODI Method [26]; the Ratio-Analysis Method, Ferguson and Sargent [26]; the Primal-Dual algorithm, Dantzig, Ford and Fulkerson [20] and Ford and Fulkerson [27]; the Decomposition techniques, Dantzig and Wolfe [21], Beale [7], Balas[33,34,35]; Branch and Bounding Methods, Land and Doig [55] etc. etc.

However, as noted by Hoffman, Mannos, Sokolowsky and Weigmann (1953) and Hoffmann (1955), most of these methods have not proved as effective in solving the general linear programming problem as the Simplex technique or some version of the Simplex technique - mainly because of the slowness in convergence and the very large number of iterations required. The Multiplex Method [30,36] though has been claimed by its author to be quite effective compared to the Simplex Method for medium sized problems. The method has, however, not been extensively coded on computers.

7. and also the Transportation Problem procedure developed by A. Henderson and R. Schlaifer [39].

8. A similar method is described in Waugh & Burrows [80].

9. It may be noted that these techniques are mainly concerned with 'special structured' linear programming problems.

9a. where the number of non-basic variables is below 300 or so.
1.1.3.1 Computer Codes. The general Simplex Method was first programmed for the SEAC Computer by Alex Orden of the U.S. Air Force and A.J. Hoffman of the National Bureau of Standards in January 1952. Later W. Orchard Hays of the Rand Corporation coded the Simplex Method for the IBM-CPC in 1952, for the IBM-701 in 1954 and the IBM-704 in 1956. Subsequently, with the tremendous growth in applications of linear programming to problems of business and industry, computer codes for the Simplex and other methods have been written on most of the intermediate and large general purpose electronic computers throughout the world.

1.2. Introduction. As stated earlier the Simplex Method (or its variations e.g. Dual Simplex Method) has thus far been the most effective and widely used general method for the solution of linear programming problems. The Simplex Method in its various forms starts initially with a basic feasible solution and continues its moves in different iterations within the feasible region till it finds the optimal solution. The only other notable variation of the Simplex Method, namely the Dual Simplex Method, on the other hand, by virtue of the special formulation of the linear programming problem, starts with an in-feasible solution and continues to move in the in-feasible region till it finds the optimal solution at which it enters the feasible region. In other respects both the Simplex and the Dual Simplex Methods follow essentially the same principle for obtaining the optimal solution. The rigorous mathematical features have been widely discussed in the literature[42, 19, 34, 35, 38, 68, 74] and only those formal aspects of this topic which are closely related...
to the subject of this thesis will be outlined.

The Multiplex Method, though reported in the literature \[30, 15, 69, 71, 27, 32]\], is not so well known and has also not been widely coded on electronic computers. It had earlier been programmed for the English Electric's Computer 'DEUCE' by the author \[72\] and the Ferranti's 'MERCURY' by Ole-John Dahl in 1960 \[15\]. Later both the above mentioned computers were obsolete and the efforts presently concentrate on coding it for UNIVAC 1100 and IBM 360. The Multiplex Method, as such, has been included in the present thesis and discussed in some detail in chapter 2. The flow diagram and the algorithm for the method is given in section 2.4, chapter 2.

The main body of the thesis consists of developing a new linear programming method which has been called the Bounding Hyperplane Method - Part I. This is explained in detail in chapter 3. The method could initially start with either a basic feasible or in-feasible point and in its subsequent moves it may either alternate between the feasible and the in-feasible regions or get restricted to either of them depending upon the problem. It is applicable as a new phase which we call phase 0 to the Simplex Method, particularly in situations where an initial basic feasible point is not available. In such cases it either results in a feasible point at the end of phase 0 or else yields a 'better' in-feasible point for phase 1 operations of the Simplex Method. Moreover, it is found that the number of

\[10\] It is abbreviated henceforth as either B.H.P. Method or B.H.P.M.
iterations required to reach either the former by the application of phase 0 or the latter by the application of first phase 0 and then phase 1 are, in general, less than those required by following phase 1 alone. This is explained with illustrations in chapter 6. Even when applied alone the method, in general, yields the optimal solution in fewer iterations as compared with the Simplex Method. This is illustrated with examples in chapter 3.

We also develop and illustrate a powerful but straightforward method whereby we first find the solution to the equality constraints and (if the former does not yield an inconsistent solution point) then the transformations to the latter are obtained from the equality solution tableau corresponding to the former. This results in reducing the iteration time appreciably for each iteration of the method. It has been called the B.H.P.M. - part II and is discussed in chapter 4.

To estimate the time taken by the B.H.P. and the Simplex Method, the two codes (written in Fortran) have been run on a number of problems taken from the literature. The results have been summarised in chapter 7.

Finally, the suggestions for further research towards (i) the extensions of B.H.P.M. to the quadratic programming problem where the function in (1.1.1) is positive semi-definite, and (ii) the accuracy of computations in linear programming, in general, are discussed in sections 8.1 and 8.2 respectively of chapter 8.
Chapter 2

THE GENERAL LINEAR PROGRAMMING PROBLEM

2. Formulation of the general linear programming problem.

2.1 Formulation 1.

Maximise the optimal function

\[ F = C X \],

subject to the conditions

\[ A X \leq P_0 \],

and

\[ X \geq 0 \],

where

\[ X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}, \quad A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ a_{21} & a_{22} & \cdots & a_{2p} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mp} \end{bmatrix}, \]

\[ P_0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_m \end{bmatrix} \],

and

\[ C = \begin{bmatrix} c_1 & c_2 & \cdots & c_p \end{bmatrix} \].

**Abbreviated as 'L.P. problem' in future discussions.**
2.2 Formulation 2.

The above can be formulated also as follows:

Maximise the optimal function (2.1.1) subject to

\[ Y - G X \]  

where

\[ 0 \leq Y \leq Q_0 \]  

\[ G = \begin{pmatrix} E_n \\ A \end{pmatrix} \]  

and \[ Q_o = \begin{pmatrix} L \\ P_0 \end{pmatrix} \]  

\( E_n \) and \( L \) being respectively the identity submatrix \((n \times n)\) and the column vector \((1 \times n)\) consisting of some very large numbers corresponding to the \(n\) elements (variables) of the column vector \(X\) \((1 \times n)\). Further \(Y\) is the column vector \((1 \times N)\)

\[ Y = \begin{pmatrix} Y^1 \\ Y^2 \end{pmatrix} \]

where

\[ Y^1 (= X) = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \]

\[ Y^2 = \begin{pmatrix} y_{n+1} \\ y_{n+2} \\ \vdots \\ y_N \end{pmatrix} \]

\[ N = n + m \]  

\( ^* \) The vectors \( Y \) and \( X \) comprise \( n \) elements (instead of \( p \)) in this formulation; also the matrix \( A \) is of the order of \((m \times n)\).
The latter formulation is used in the Multiplex Method [30] for finding the optimal solution to the general linear programming problem. We shall now discuss the Multiplex Method and return to the former formulation for describing the B.H.P. Method and the related work in subsequent chapters.

2.3 The Multiplex Method of Linear Programming.

Let the inequalities (2.2.2), in general, be considered as follows

\[ Y \leq \gamma \leq Y \quad , \quad (2.2.8) \]

where \( Y \) and \( \bar{Y} \) denote the lower and upper bounds of \( Y \). Let \( I, J \) be the set of indices of the rows and columns of \( G (= g_{ij} \), an \( N \times n \) matrix). Also let \( G_i, (i \in I) \) define the row space of \( G \) so that

\[ G_i = [g_{i1} \ g_{i2} \ldots g_{in}] \quad , \quad i \in I \quad . \quad (2.2.9) \]

Further let \( I_{\text{ind}} \subset I \), \( I_{\text{dep}} \subset I \) be the set of indices of the rows of \( G \) corresponding to the matrices \( E_n \) and \( A \) respectively. \( I_{\text{ind}} \) and \( I_{\text{dep}} \) then define the 'independent' and 'dependent' sets (of the variables) \( Y^1 \) and \( Y^2 \) respectively.

Let \( I_{\text{opm}} \subset I \) be the set of indices \( i \) for which the elements \( Y_i, i \in I_{\text{opm}} \) belong to the operation vector \( \hat{Y} (= 1 \times \nu, 0 \leq \nu) \), that is

\[ \hat{Y} = [\hat{Y}_i] \quad , \quad i \in I_{\text{opm}} \quad . \quad (2.2.10 \text{(i)}) \]

\(^{13}\text{Frisch [29, 30] calls this vector the operation set. Initially } \hat{Y} \text{ is a null vector.}\)
The vector \( \hat{Y} \) is such that
(i) the elements (variables) \( \hat{y}_i, i \in I_{om} \) are at their either lower or upper bounds as the case may be and are linearly independent,
and (ii) if \( \hat{y}_i - \bar{y}_i \) then \( \rho_i < 0 \)
or else if \( \hat{y}_i = \bar{y}_i \) then \( \rho_i > 0 \)
where \( \rho_i \), called the regression coefficients, are the element of the regression vector

\[
R = |\rho_i|, i \in I_{om}, \tag{2.2.11}
\]
given by

\[
R = M^{-1} M_{oh}, \tag{2.2.12}
\]

The moment matrix \( M \) and the column vector \( M_{oh} \)
are given by

\[
M = \left( m_{i,h} \right), \quad i, h \in I_{om}, \tag{2.2.13}
\]

\[
M_{oh} = G_h C^T \tag{2.2.14}
\]

where

\[
m_{i,h} = G_i G_h^T, \quad i, h \in I_{om}, \tag{2.2.15}
\]

The moment matrix \( M \) is a symmetric matrix in view of (2.2.15) and so also is the inverse moment matrix \( M^{-1} \).
and the superscript $^T$ denotes the transposition to a column vector.

The condition (2.2.10 (ii)) above is termed the sign-correctness of the regression coefficients; if the opposite is the case then the latter are said to be sign-incorrect.

Additionally, in view of (2.2.10 (ii)) above the direction numbers of $\gamma_i$, $i \in I_{obm}$ are set to zero, so as to retain them on their bounds. The direction numbers denoted by $d_i$, $i \in I$ are given by the vector

$$D = (D^1, D^2) = |d_i|$$

where

$$D^1 = |d_i^1|, \quad D^2 = |d_i^2|$$

and

$$d_i^1 \begin{bmatrix} c_i + \sum_{k \in I_{obm}} g_{ki} / \rho_k \\ = 0, \quad i \in (I_{obm} \cap I_{ind}) \end{bmatrix}$$

$$d_i^2 \begin{bmatrix} \sum_{j \in J} g_{ij} + \sum_{k \in I_{obm}} g_{ij} / \rho_k \rho_{kj} \\ = 0, \quad i \in (I_{obm} \cap I_{dep}) \end{bmatrix}$$

These direction numbers, in turn, are utilised in selecting $\gamma \eta$, $\eta = i \in I$ for its entry in the operation vector by

15. The superscript $^T$ would henceforth be used to denote the transposition of either a matrix or a row to column vector and vice-versa, as the case may be.
following the criterion below,

$$\lambda_{\text{min}} = \frac{\Delta y^i}{d \eta} = \min_{i \in I'} \left( \frac{\Delta y^i}{d_i} \right),$$  \hspace{1cm} (2.2.20)

where

$$\Delta y^i = \begin{cases} y^i - y^o_i & \text{for } d_i < 0 \\ y^o_i - y^i & \text{for } d_i > 0 \end{cases},$$  \hspace{1cm} (2.2.21)

$y^o_i$ are the elements of the vector

$$Y^o = \left| y^o_i \right|.$$  \hspace{1cm} (2.2.22)

the superscript 'o' denoting the initial point which is either readily available from the problem itself (for example, in many situations it could be the lower bound values of the variables) or can be found and $I'$ is defined by

$$I' \cup I_{\text{on}} = I.$$  \hspace{1cm} (2.2.23)

$\lambda_{\text{min}}$ and $y^o$ are termed the breaking out parameter and the variable respectively; the latter is included in $\hat{Y}$. The new inverse moment matrix required for computing the regression vector $R$ is then obtained from

$$M_{\text{new}}^{-1} = \begin{bmatrix} M' & Z \\ Z^T & 1/\omega \eta \end{bmatrix}.$$  \hspace{1cm} (2.2.24)

As it happens, in most of the macro-economic problems, to which the Multiplex Method has been mainly applied, an initial point is generally available; however, if it is not available then it can be found using one of the methods discussed in [30].
where
\[ M' = M_{\text{old}}^{-1} - Q Z^T, \] (2.25)
and the column vectors \( Q, \mu, Z \) and the scalar \( \omega_{\eta \eta} \) are given by
\[ Q = M_{\text{old}}^{-1} \mu; \quad \mu = \begin{bmatrix} m_{\eta} \end{bmatrix}; \quad \omega_{\eta \eta} = m_{\eta} + Q^T \mu. \] (2.26)
\[ Z = Q / \omega_{\eta \eta}; \quad \omega_{\eta \eta} = m_{\eta} + Q^T \mu. \] (2.27)

Since the regression coefficients \( \beta_i, i \in I_{\text{efm}} \) should remain sign correct, after \( y \) has also entered the operation vector \( \hat{\gamma} \) it is checked that the conditions (2.2.10) are satisfied. If, therefore, one or more variables turn out to be sign-incorrect then the one encountered last as sign-incorrect is removed from \( \hat{\gamma} \). As often happens, \( \hat{\gamma} \) may no longer consist of sign-incorrect variables; but if this is not the case then we continue to remove, one by one, sign-incorrect variables till \( \hat{\gamma} \) comprises only sign-correct variables. The elements of the inverse moment matrix are updated each time a sign-incorrect variable is removed from \( \hat{\gamma} \) and the new inverse moment matrix elements are obtained (each time) from,
\[ M_{\text{new}}^{-1} = M_{\text{old}}^{-1} - S \cdot \frac{S^T}{\xi_{\gamma \gamma}}, \] (2.28)
where \( \gamma \) is the index of the variable \( y_{\gamma} \) turning out to be sign-incorrect, \( \xi_{\gamma \gamma} \) is the pivotal element of the old inverse moment matrix and \( S ( = | \xi_{i \gamma} | = | \xi_{\gamma j} | ; \; i, j \in I_{\text{efm}} ) \) is the \( \gamma \text{th} \) column (or row) vector of the old inverse moment matrix. In computing \( M_{\text{new}}^{-1} \) from (2.2.28) the elements of those vectors which correspond to the \( \gamma \text{th} \) row and column turn out to be null.
vectors and hence care is taken to squeeze the matrix correspondingly by one dimension each time a sign-incorrect variable is removed from $\hat{Y} \cdot J$.

The new point at the end of each iteration is given by

$$\hat{Y} = Y^o + \lambda_{\min} \cdot D^T$$

and this, in turn, becomes the starting point for subsequent iterations.

If, in the course of different iterations, it is found that

$$((\forall i \in I)(a_i = 0))$$

then it is implied that the optimal point has been obtained. The general optimal criterion (which would also be satisfied if (2.2.30) is true), however, is that the preference direction number

$$d_0 < \varepsilon$$

where

$$d_0 = D^T C^T$$

and $\varepsilon$ is the threshold value designed to take into consideration the round off errors as well as errors in the data. The optimal value when (2.2.31) is true is given by

$$F = C Y^1 = C X.$$  

At this stage, if

$$\nu = \eta \quad (\text{refer 2.2.10(i)})$$

then the optimum point obtained is unique. If, on the other hand, 

$$\nu < \eta \quad ,$$

it implies that there exists at least one set of $\eta - \nu$ linearly
independent variables which generate a linear manifold such that the value of the optimal function is the same at every point of this manifold.
2.4. The Multiplex flow diagram and the algorithm.

2.4.1. The flow diagram.

Start:

Input data, storing of a large number and also the threshold value $\varepsilon$.

Compute the direction numbers for all the variables and the preference direction number.

Is the preference direction number less than the threshold value? [Yes/No]

- Yes: Print the values of the variables, the shadow prices (vector $R$) and the optimal function value.

- No: Compute the breaking out parameter and the values of the variables in the new point; find the breaking out variable (for its inclusion in the operation vector) and set its direction number to zero.

Update the operation vector, compute the new inverse moment matrix and the regression vector.

Have all the variables sign-correct regression coefficients? [Yes/No]

- Yes: Start

- No: Find the index of the last sign-incorrect (if more than one) variable for its exclusion from the operation vector.
2.4.2 The algorithm. For the purpose of describing the algorithm of this section let $M$ denote the section 2.4.2. The algorithm is iterative and detailed as follows.

$M$. Find $D$ and $d_\circ$ from (2.2.16) to (2.2.32) respectively.

M.1 Test $d_\circ$:

M.1.1 if $d_\circ < \varepsilon$, then print the values of all the variables, the shadow prices and the optimal function value given by (2.2.5) and (2.2.33) respectively.

M.1.2 if not, then

M.1.2.1 (i) compute $\lambda_{\min}$, $\eta$ from (2.2.20), $Y$ from (2.2.29) and set $d_{\eta} = 0$,

M.1.2.2 (ii) compute the new inverse moment matrix from (2.2.24) or (2.2.28) - as the case may be, the regression vector $R$ from (2.2.11) and update the operation vector.

M.2 Test $\rho_i$, $i \in I_{\text{obj}}$:

M.2.1 if $\rho_i > 0$ test $\hat{q}_{i}^{-}$ and count $i$ in $q_{i}^{-}$:

M.2.1.1 if $\hat{q}_{i} = \bar{q}_{i}$, $\rho_i$ is sign-correct, go to M.3;

M.2.1.2 otherwise $\rho_i$ is sign-incorrect, go to M.2.2.3;

M.2.2 if not, test $q_{i}^{-}$ and count $i$ in $q_{i}^{-}$:

M.2.2.1 if $\hat{q}_{i} = q_{i}$, $\rho_i$ is sign-correct, go to M.3;
M.2.2.2 otherwise $p_i$ is sign-incorrect;

M.2.2.3 $i = \gamma$, go to M.1.2.2;

M.3 test $(q_1 + q_2)$:

M.3.1 if $(q_1 + q_2) < \omega$ go to M.2;

M.3.2 otherwise go to M.

\* $\gamma$ is the last sign-incorrect variable encountered in $\gamma$. 
Chapter 3

THE BOUNDING HYPERPLANE METHOD OF LINEAR PROGRAMMING - PART I

Let the general l.p. programming be considered as formulated in section 2.1 of chapter 2. We consider the inequality constraints of the system (2.1.1)-(2.1.3) alone in this chapter; the equality constraints can be treated independently of the former and hence are treated separately as part II of the B.H.P. Method in the next chapter.

3.1 The Inequalities. Let the inequality constraints in (2.1.2) be converted to equations by (i) multiplying the constraints of ' \geq ' type (I), if any, by -1 so as to convert them first to constraints of ' \leq ' type (II) and then (ii) adding a positive slack variable to each of the type (II) constraints. The resulting system of equations and the l.p. problem could then be rewritten as

Maximise

\[ F = C \times X, \]

subject to

\[ B \times X = P, \]

and

\[ X \geq 0, \]

where

\[ X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}. \]
$$B = \begin{vmatrix} p_1 & p_2 & \cdots & p_n \end{vmatrix} = \begin{vmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{vmatrix} , \quad (3.1.5)$$

$$C = \begin{vmatrix} c_1 & c_2 & \cdots & c_p & c_{p+1} & c_{p+2} & \cdots & c_n \end{vmatrix} , \quad c_j \begin{cases} = c_j & \text{for } j = 1, 2, \ldots, p \\ = 0 & \text{for } j = p+1, p+2, \ldots, n. \end{cases} \quad (3.1.6)$$

$$\mathcal{P}_0$$ is given by (2.1.5) with signs corresponding to type II inequalities, if any, changed, $0$ is a null vector and

$$n = p + m . \quad \text{(3.1.6 \, (i))}$$

If we consider the slack variables in the set of equations (3.1.2) we note that they, being a set of $m$ linearly independent vectors corresponding to $m$ structural constraints in the original set (2.1.2), constitute for our study an initial basis. Let $P_{p+1}, P_{p+2}, \ldots, P_n$ denote this set of $m$ linearly independent vectors so that the initial basis is given by

$$\beta = \begin{vmatrix} p_{p+1} & p_{p+2} & \cdots & p_n \end{vmatrix} . \quad (3.1.7)$$
whence we have,

\[ \overline{X} = \beta^{-1} P_0 \quad \text{where} \quad \overline{X} = \begin{bmatrix} \overline{x}_{p+1} \\ \vdots \\ \overline{x}_{p+m} \end{bmatrix} \]  
\[ \overline{B} = [\overline{B}_1, \overline{B}_2, \ldots, \overline{B}_n] \]
\[ = [\beta'^{P_1}, \beta'^{P_2}, \ldots, \beta'^{P_n}] \]
\[ = \beta'^{-1} P_1 P_2 \cdots P_n \]
\[ = \beta'^{-1} B \]  
\[ F = C \times \]
\[ = \overline{C} \overline{X} \]
\[ = \overline{C} \beta'^{-1} P_0 \quad \text{where} \quad \overline{C} = [c_{p+1}, c_{p+2}, \ldots, c_n] \]  
\[ Z = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_n \end{bmatrix} \]
\[ = \begin{bmatrix} z_1 - c_1 \\ z_2 - c_2 \\ \vdots \\ z_n - c_n \end{bmatrix} \]
\[ = [\overline{C} \overline{B}_1 - c_1, \overline{C} \overline{B}_2 - c_2, \ldots, \overline{C} \overline{B}_n - c_n] \]
\[ = [\overline{C} \beta'^{-1} P_1 - c_1, \overline{C} \beta'^{-1} P_2 - c_2, \ldots, \overline{C} \beta'^{-1} P_n - c_n] \]
\[ = \overline{C} \beta'^{-1} B - C \]  
\[ (3.1.11 \text{ (ii)}) \]

In particular the initial solution \( \overline{X} \) given by (3.1.8) may thus be either an infeasible or a feasible solution. It need not necessarily be the latter as is prerequisite with some other methods, for example the Simplex Method. The '−' sign associated with \( \overline{X}, \overline{B}, \overline{C} \) corresponds to the current point or tableau.
Let the general B.H.P. (method) tableau\(^*\) be given by

\[
    \begin{array}{c|c}
        \mathbf{c} & \mathbf{0} \\
        \hline
        \mathbf{B} & \mathbf{X} \\
        \hline
        \mathbf{Z} & \mathbf{F}
    \end{array}
\]  

(3.1.12)

where the contents of the tableau are defined by (3.1.8) to
(3.1.11). Let \( I, J \) be the set of indices of the rows and
columns of \( \mathbf{B} \). Let \( I_1 \subset I \) denote the set of indices \( i \) for which

\[ x_i < 0, \ i \in I_1 \text{ (if any)}. \]  

(3.1.13)

Let \( J_1 \subset J \) be the set of indices \( j \) for which

\[ Z_j < 0, \ j \in J_1. \]  

(3.1.14)

and let \( \mathcal{A} \) and \( \text{mod} Z_j, i \in J_1 \) respectively denote the space spanned by
\( P_j, \ j \in J_1 \) and the hyperplane with all \( Z_j \) \( (i \in J_1) \)
coefficients positive. Also let \( \mathbf{B}_i \ (i \in I) \) define the row
space of \( \mathbf{B} \) so that,

\[
    \mathbf{B}_i = | b_{i1} b_{i2} \ldots b_{in} |, \ i \in I
\]  

(3.1.15)

and let \( \mathbf{B}_j, \ (j \in J_1, \ i \in I) \) denote the corresponding row
elements of \( \mathbf{B}_i \ (i \in I) \) for which (3.1.14) holds. Further let
\( I_2 \subset I \) be the set of indices \( i \) for which

\[ s_i < 0 \]  

(3.1.16)

where\(^{20}\)

\[ s_i = \sum_j \mathbf{B}_i^j Z_j, \ j \in J_1. \]  

(3.1.17)

\(^{14}\) The general tableau \( \mathbf{M} \) is different from the one considered
in the Simplex Method since the negative elements in \( P \) are re-
tained under \( X \).

\(^{20}\) "\( s_i \)" is proportional to the cosine of the angle between the
normals to the \( \text{mod} Z_j \) hyperplane (in the direction away from the
origin) and the \( i^{th} \) hyperplane (infeasible direction). Geometric-
ically, by considering (3.1.16) we therefore look for those hyper-
planes that have this angle less than 90 degrees; hence, the name
'bounding' for such hyperplanes.
I_2 then defines, what from now on will be termed as, the set of the bounding hyperplanes. We next define a bounding vector \( \Delta \) comprising of elements \( \Delta_i \ (i \in I_2) \) such that,

\[
\Delta_i = \left| \frac{x_i}{e_i} \right| , \ i \in I_2 .
\]  

(3.1.18)

Geometrically \( \Delta_i \) may be interpreted as a scalar which is proportional to \( d_i \), that is

\[
\Delta_i = k \ d_i , \ i \in I_2 ,
\]

(3.1.19)

where \( d_i \) denotes the distance from the origin of the bounding hyperplane in the increasing direction of the normal to the mod \( Z_j \ (j \in J_1) \) plane and \( k \) is a constant (in so far as the current solution point is concerned), given by

\[
k = \left( \sum_j Z_j^2 \right)^{-\frac{1}{2}} , \ j \in J_1 .
\]

(3.1.20)

The bounding vector \( \Delta \) thus is a function of the distance of the bounding hyperplanes \( i \in I_2 \), and the particular hyperplane \( \gamma \), for which

\[
\Delta_\gamma = \min_i (\Delta_i) , \ i \in I_2
\]

(3.1.21)

defines, therefore, the nearest bounding hyperplane in the increasing direction of the normal to the mod \( Z_j \ (j \in J_1) \) plane. Clearly the optimum point, if \( x_\gamma > 0 \), must either lie
on or below the hyperplane \( \eta \) given by (3.1.21), that is, in that half space of the \( \eta \) hyperplane which is towards the origin; similarly, if \( x_\eta < 0 \), then the optimum point must either lie on or above the hyperplane \( \eta \), that is, in that half space of the \( \eta \) hyperplane which is away from the origin.

Utilising the above properties we select in each iteration the \( \eta \) hyperplane out of the set of the bounding hyperplanes as observed from the basis point (that is the new origin) reached at the last iteration until we arrive at the optimal solution to the problem by following the decision rules enumerated below.

### 3.2 Decision rules; class R and S.

#### Class R.

If

\[
(\exists j \in J) (Z_j < 0)
\]  (3.2.1)

then the rules are defined as follows:

**Rule R1.** Select \( i (=\eta) \in I_2 \) from (3.1.21) above. This may be termed the Exit Criterion for the Class R.

**Rule R2.** If \( F^* \) denotes the value of the objective function at the new point then, for \( i (=\eta) \in I_2 \) and \( j \in J_i \),

\[
F^* = F - x_i \left( \frac{Z_j}{\bar{B}_i} \right)
\]  (3.2.2)

or

\[
\Delta F = F^* - F = -x_i \left( \frac{Z_j}{\bar{B}_i} \right);
\]  (3.2.3)

where the particular \( j \in J_i \) is yet to be specified and measures the change in the optimal function value from one to the next move. Since we consider infeasible solutions as well in our search for the optimum, the following two possibilities exist -
Either (i) \( x_\eta > 0 \) \hspace{1cm} (3.2.4)

or (ii) \( x_\eta < 0 \). \hspace{1cm} (3.2.5)

R2.1. If (3.2.4) is true, select \( T = j \in J \) from

\[
\max_j |Z_j / B_i^j|, \quad \text{if } i = \eta \in I_2 \hspace{1cm} (3.2.6)
\]

which gives the maximum increase to the optimal function value.

Alternatively \( T = j \in J \) could also be selected from

either (i) \( \min_j |Z_j / B_i^j| \) or (ii) \( \max_j (Z_j B_i^j) \),

where in both cases \( B_i^j > 0, \ j \in J, \ i = \eta \in I_2 \).

R2.2. If (3.2.5) is true then select \( T = j \in J \), for which

\[
\max_j (Z_j / B_i^j), \quad B_i^j < 0, \ j \in J, \ i = \eta \in I_2 \ .
\]

however, if it happens that

\[
((\forall j \in J) (B_i^j > 0)) \hspace{1cm} (3.2.7)
\]

(so that \( T \) satisfying (3.2.7) can not be found) then select \( T \) from

\[
\min_j |Z_j / B_i^j|, \quad B_i^j > 0, \ j \in J, \ i = \eta \in I_2 \ .
\]

If a tie occurs in the selection of \( T \) then we choose any of

the \( j \) for which this tie occurs. This is called the Entry

Criterion for class \( R \).

In the former case the value of the optimal function increases as compared with its value in the last iteration but if the
latter were to be true then the value of the optimal function decreases in the next iteration. (This might appear to lead to a situation where an old basis may repeat itself. However, by deciding to move in each iteration on the (nearest) bounding hyperplane alone in the $\mathcal{H}$ space, we believe it is highly unlikely that such a situation can arise. This has been confirmed by the empirical evidence of running the computer program (for B.H.P. method) on a number of both the small and large examples and no case has thus far been encountered where the algorithm does not terminate.)

The above decision rules thus uniquely determine the hyperplane $\eta_j$ to which to move and the axis $\tau_j$ along which to move to the former. In particular $b_{\eta_j \tau_j}$ determines the pivotal element on which the Gaussian eliminational transformations, according to formulae given in section 3.2.1, are performed to obtain the new point and the tableau.

The foregoing rules will therefore either lead to the optimum point wherein

$$((\forall j \in J)(Z_j > 0) \land (\forall i \in I)(x_i \geq 0));$$ (3.2.10)

or else a point will be reached which is considered in class $S$ below.

Class $S$. If (refer also the Dual Simplex Method [56] )

$$((\exists i \in I)(x_i < 0) \land (\forall j \in J)(Z_j \geq 0));$$ (3.2.11)

then the rules are defined as follows.
Rule S1. Select \( t(-\eta') \in I_4 \) for which

\[
\Delta'_{\eta'} = \max_{i} (\Delta'_i) , \ i \in I_4 , \quad (3.2.12)
\]

where

\[
\Delta'_i = \left| \frac{x_i}{s_i} \right| , \ \delta'_i \neq 0 , \ i \in I_4 , \quad (3.2.12 \text{ (i)})
\]

\[
\delta'_i = \sum_i b_{ij} Z_j , \ j \in J , \ i \in I_4 , \quad (3.2.12 \text{ (ii)})
\]

and \( \eta' \) denotes that hyperplane amongst the infeasible hyperplanes which is farthest from the origin in the decreasing direction this time of the normal to the mod \( Z_j (j \in J) \) hyperplane for the optimum point must now either lie on or above the \( \eta' \) hyperplane given by (3.2.12). If, however, it is found that

\[
\left( (\forall i \in I_4) (\delta_i = 0) \right) \quad (3.2.13)
\]

then we select the \( \eta' \) hyperplane from

\[
\max_{i} \left| \frac{x_i}{\sqrt{\sum_j b_{ij}^2}} \right| , \ j \in J , \ i \in I_4 \quad (3.2.14)
\]

The geometrical interpretation of \( \Delta'_i \), \( \delta'_i \) and \( \eta' \) are analagous to those of \( \Delta_i \), \( \delta_i \) and \( \eta \) discussed earlier with the above mentioned restrictions that are peculiar to the situation in class S. This is called the Exit Criterion for the class S.

Rule S2. Select \( \gamma' = j \in J_2 \) for which

\[
\max_{j} \left( Z_j \sqrt{b_{jj}} \right) , \ j \in J_2 , \quad \eta' \in (\epsilon I_2) , \quad (3.2.15)
\]
where \( J_2 \) denotes the set of indices \( j \in J \) for which
\[
\bar{v}_{ij} < 0, \ j \in J, \ i = \eta \in \mathbb{I}_J
\]  
(3.2.16)

This is called the Entry Criterion for the class \( S_i \), and is the same as in the Dual Simplex Method [36]. (A tie, if any, is handled as in the case R2.2).

The optimal function value in class \( S \) decreases in each move and must in a finite number of steps yield the optimal solution, if it exists, satisfying (3.2.10).

3.2.1 The pivotal transformations. The pivotal transformations, for obtaining the new point and the tableau for either of the above two classes are given by
\[
\bar{v}_{ij}' = \bar{v}_{ij} - \left( \frac{\bar{v}_{ij}}{\bar{v}_{ij}'} \right) \bar{v}_{ij}', \ j \in J_1,
\]
(3.2.17)
and
\[
\bar{v}_{ij}' = \bar{v}_{ij}' \left( \frac{\bar{v}_{ij}'}{\bar{v}_{ij}'} \right), \ i = \eta
\]
where a prime denotes the new values of the elements in the transformed tableau. (The new values of the objective function, the shadow prices and the solution vector are respectively given by,
\[
\begin{align*}
\bar{v}_{ij}' &= \bar{v}_{ij} + \bar{v}_{m+i} = \bar{v}'_i, \\
\bar{v}_{ij}' &= \bar{v}_{m+i,j} = \bar{v}'(= \bar{v}'_i), \ j \in J, \\
\bar{v}_{ij}' &= \bar{v}_{i,n+j} = \bar{v}'(= \bar{v}'_i), \ i \in I.
\end{align*}
\]  
(3.2.18)

3.3 The unbounded and non-feasible problems.

The condition
\[
((\forall i \in I) ((\forall j \in J_i) (\sum \bar{B}_i |Z_j| < 0)))
\]  
(3.3.1)

20(e) \( \eta = \eta' \) in (3.2.17), if the pivotal row (hyperplane) is selected following class \( S \) decision rules.
which could also be written as

\[
((\forall i \in I)(\forall j \in J_i)((\sum_i B_{i,j}^t Z_j \geq 0))) ,
\]

(3.3.1 (i))
or

\[
((\forall i \in I)(\delta_i > 0)), \delta_i \text{ given by (3.1.17)}, \quad (3.3.1 (ii))
\]

implies the unbounded problem; and the condition

\[
((\exists i \in I)(x_i < 0) \& (\forall j \in J_i)(\bar{b}_{i,j} > 0))
\]

(3.3.2)
implies the non-feasible problem.

In either case there is no optimal solution to the problem.

The Unbounded Problem. Consider the space \( \mathcal{H} \) spanned by \( \mathcal{P}_j, j \in J_i \) for which (3.1.14) holds. Then if \( \theta_i, i \in I \) denotes the angle between the normals in \( \mathcal{H} \) to the mod \( Z_j \) hyperplane (in the direction towards the origin) and the \( \mathcal{H} \) hyperplane (in the feasible direction) we have

\[
\theta_i = \cos^{-1}\left( q_i \sum_j (\bar{B}_{i,j}^t |Z_j|) \right), i \in I, j \in J_i, \quad (3.3.3)
\]

where \( q_i \) is a positive number given by

\[
q_i = k \left( \sum_j (\bar{B}_{i,j}^t)^2 \right)^{-1/2}, i \in J_i, j \in I. \quad (3.3.4)
\]

If (3.3.1) or (3.3.1(ii)) holds for each \( i \) then (3.3.3) indicates the non-existence of any bounding hyperplanes (or in other words it indicates the existence of an open convex solution set). The value of the optimal function can thus be increased infinitely in the increasing direction of the optimal function and no finite (maximal) optimal solution to the problems exists.

The non-feasible problem, (3.3.2) implies that (2.1.3) cannot be satisfied and hence no optimal solution to the problem exists.
3.4 Degeneracy and Cycling.

Let us reconsider the decision rules for the classes R and S. In following the exit criteria in either of the two classes, if the \( \gamma \) hyperplane happened to be the one with \( x_\gamma = 0 \) then the new solution would be degenerate (a tie in computing \( \max \Delta_i \) or \( \min \Delta_i \) from (3.1.21), even if the current solution is non-degenerate, may lead to degeneracy in the new solution). It must be significantly noted at this stage that those basic variables which have values equal to zero but correspond to non-bounding hyperplanes are, in any case, excluded from further consideration as we select only bounding hyperplanes in sub-space \( \mathcal{A}^* \). In effect, therefore, we are left with only such cases for which

\[
\begin{align*}
\xi_i &= 0, \\ i &\in I_2^d
\end{align*}
\]  

(3.4.1)

where \( I_2^d \subseteq I_2 \) denotes the set of indices \( i \) for which either \( \xi_i = 0 \), in degenerate cases, or \( \Delta_i = \text{constant}, \; i \in I_2 \) in non-degenerate cases.

In such a situation the \( \gamma \) hyperplane may, in general, correspond to any one of the \( \xi_i \) for which (3.4.2) holds. We would, however, proceed as follows.

Since \( \Delta_i = 0, \; i \in I_2^d \), (that is, all such hyperplanes are equidistant from the origin in the increasing direction of the
normal to the mod $Z_j \ (j \in J_1)$ plane, we determine the most convergent hyperplane from

$$\Theta_{\eta} = \min_{i} (\Theta_i), \ i \in I_2^d.$$  \hspace{1cm} (3.4.3)

If there is still a tie in computing the $\eta$ hyperplane uniquely from (3.4.3) then we determine it from

$$\Theta'_{\eta} = \min_{i} (\Theta'_i), \ i \in I_2^d;$$  \hspace{1cm} (3.4.4)

where

$$\Theta'_i = \cos^{-1} \left( \frac{q'_i \sum_j (\bar{b}_{ij} z_{ij})}{\sqrt{\sum_{j} (\bar{b}_{ij} z_{ij})^2}} \right), \ j \in J,$$  \hspace{1cm} (3.4.5)

and

$$q'_i = \left( \sum_j z_{ij}^2 \right)^{\frac{1}{2}} \left( \sum_j \bar{b}_{ij}^2 \right)^{-\frac{1}{2}}, \ j \in J.$$

It is indeed highly improbable that a tie could still persist in selecting the $\eta$ hyperplane from either (3.4.3) or (3.4.4) because if for no other reason, rounding errors are introduced during the computations. For this reason and consequent saving
Figure 3.1
in computing time we find that a simpler criterion for selecting the \( \eta \) hyperplane from
\[
\delta_\eta = \max_{i} |\delta_i|, \; i \in I_d^d,
\]
is efficient and expedient in resolving both the degenerate and/or non-degenerate ties. As we shall show in illustrations given in section 3.7, we effectively resolve the cycling phenomenon in an artificially constructed example (for the Simplex Method) using the Criterion (3.4.6). It may also be stated here that the same criterion has been used for resolving the cycling phenomenon in another example due to Hoffman [41]. The latter is, however, not reported in the illustrations of section 3.7.

3.5 Geometrical Interpretation of the Method.

Let us consider the following two dimensional l.p. problem.

Maximise
\[
F = c_1 x_1 + c_2 x_2
\]
subject to
\[
\alpha_{i1} x_1 + \alpha_{i2} x_2 \leq b_i, \; i = 1, 2, \ldots, m,
\]
and
\[
x_1, x_2 \geq 0.
\]
where \( b_i, \; i = 1, 2, \ldots, m \) are unrestricted in sign. We presume without loss of generality that \( c_1, c_2 > 0 \). Let the \((m+2)\) constraints (3.5.2) and (3.5.3) be as shown in the graph of Fig. 3.1. The shaded region \( K \) formed by them defines the feasible region as against the infeasible region surrounding \( K \). Let the optimal function
be as shown by the dotted line in the diagram. The optimum point is obtained by moving the optimal function parallel to itself until it reaches the point \( M (L) \) where it attains the maximum (minimum) value.

As we see from the diagram in figure 51, certain half-planes converge towards the maximal optimal point \( M \) and certain others similarly converge towards the minimal optimal point \( L \). (It may be noted that the \( 1^{st} \) and \( 2^{nd} \) half-planes are 'completely' redundant ones and although they appear to converge towards \( M \) and \( L \) respectively; they, in fact, are the least helpful in leading towards the optimum point). Since as in a maximising case we are interested in reaching the point \( M \) it will obviously be advantageous to search for the former so that by moving in different iterations on one or more of these half-plane we ultimately converge at the point \( M \). To distinguish the two types of the half-planes, we consider the angles (denoted by \( \theta_i \), \( i = 1, 2, \ldots, m \)) between the normal in the increasing direction of the functional \( Z = -c_1 x_1 - c_2 x_2 \) and the normals to the half-planes given by (3.5.2) in the feasible region. The bounding half-planes that converge towards \( M \) are those for which

\[
0^\circ \leq \theta_i \leq 90^\circ \quad (3.5.4)
\]

20º This is same as the decreasing direction of the mod \( Z_j \) plane (that is towards the origin).
Similarly the non-bounding half-planes that converge towards \( L \) are the ones for which

\[
90^\circ \leq \theta_i \leq 180^\circ \quad (3.5.5)
\]

From analytical geometry we know that the cosine of the angle (as defined above) between the \( \# \) half-plane \( a_i \xi_1 + a_{i2} \xi_2 \leq b_i \) \((i = 1, 2, \ldots, m)\) and the functional \( Z = -c_1 \xi_1 - c_2 \xi_2 \) is given by

\[
\cos \theta_i = \frac{-a_i c_1 - a_{i2} c_2}{\sqrt{a_i^2 + a_{i2}^2} \sqrt{c_1^2 + c_2^2}} \quad (3.5.6)
\]

\[
= k_i (-a_i c_1 - a_{i2} c_2) \quad (3.5.7)
\]

where \( k_i \) is a positive number given by

\[
k_i = \left( a_i^2 + a_{i2}^2 \right)^{\frac{1}{2}} (c_1^2 + c_2^2)^{-\frac{1}{2}}. \quad (3.5.8)
\]

In our search for the optimum we, however, consider (refer discussion \( \omega (3.1.19) \)) the increasing direction of the normal to the mod \( Z_j \) \((= c_1 \xi_1 + c_2 \xi_2)\) plane instead so that if \( \omega \) may define \( \phi_i \) to be the angle between the former and the normal to the \( \# \) half-plane in the feasible direction then we observe that
\( \phi_i \) is related to \( \theta_i \) by

\[
\phi_i + \theta_i = 180^\circ
\]  

(3.5.9)

whence, in view of (3.5.4), (3.5.5) and (3.5.7), the type of the \( i^{th} \) half-plane as bounding or not is determined from

\[
-\alpha_i \ c_1 - \alpha_{i2} \ c_2 < 0
\]  

(3.5.10)

or

\[
-\alpha_i \ c_1 - \alpha_{i2} \ c_2 \geq 0
\]  

(3.5.11)

respectively. Computationally, therefore, we determine the bounding nature of the \( i^{th} \) hyperplane from the negativity of the inner product given by the left hand side term of (3.5.10).

We next wish to determine the specific half-plane out of all the bounding half-planes in the system which leads us towards the optimum point \( M \) such that the 'completely' redundant \( \left( m-1 \right)^{th} \) half-plane is avoided too. Let us at this stage, for simplicity of exposition consider just \( m = 9 \) half-planes with optimal function as shown by the dotted line. To make the figure (problem) representative for all the types of half-planes we consider the nine half-planes to comprise four non-bounding and five bounding half-planes. These are numbered 1, 2, 3, 9 and 4, 5, 6, 7, 8 respectively in fig.3.1. Furthermore, of the former number 1 is a redundant non-bounding half-plane and of the latter, numbers 7 and 8 are redundant bounding half-planes. (The redundant half-planes 1, 7 and 8, as may be seen from figure3.1, are such
that they play no part in defining the feasible region $K$.

The optimal point $M$ is defined by the intersection of two bounding half-plane numbers 4 and 6. If we now look at the increasing direction of the normal to the mod $Z$ plane (indicated by the thick line with an arrow) in figure 31 then we find that the half-plane number 4 happens to be one of these two half-planes; in fact, it is also the nearest (out of all the bounding half-planes) one from the origin $O$. Thus if we know the distances of all the bounding half-planes in the increasing direction of the normal to the mod $Z$ plane then by selecting the nearest (from the origin) half-plane we are able to determine a plane which meets our requirements. In the present example it, in fact, passes through the optimum point $M$ also.

For this purpose, we need, however, not determine the exact distances of all the half-planes. Let us examine the 'static' $\mathcal{S}_i$, $i = 1, 2, \ldots, m$ given by

$$\mathcal{S}_i = \frac{b_i}{\left(-a_{i1} c_1 - a_{i2} c_2\right)} \quad (3.5.12)$$

$$= \frac{b_i}{\sqrt{a_{i1}^2 + a_{i2}^2}} \cos \theta_i$$

$$= \left(c_1^2 + c_2^2\right)^{-1/2} \left(\frac{b_i}{\cos(\theta_i)}\right)$$

$$= k \cdot d_i \quad (3.5.13)$$

where $k$ is a constant given by

$$k = \left(c_1^2 + c_2^2\right)^{-1/2} \quad (3.5.14)$$
and \( p_i \) and \( d_i \) respectively are (refer figure 3.2) the perpendicular distance and the distance in the increasing direction of the normal to the mod \( Z_i \) half-plane of the \( \frac{1}{A} \) half-plane from the origin. It may significantly be noted that the latter is directly proportional to \( d_i \) given by (3.5.12), hence, in the above example, we choose the nearest half-plane number 4 from

\[
\min \left( d_i \right), \ i = 4, 5, \ldots, 8.
\]

The half-plane 4 is found to possess another interesting property namely, if we consider the angle \( \theta_i \) given by (3.5.4) for the four bounding half-planes then

\[
\begin{align*}
\theta_A &= \min_i (\theta_i), \ i = 4, 5, 6, 7; \\
\phi_A &= \max_i (\phi_i), \ i = 4, 5, 6, 7.
\end{align*}
\]

(Also, \( \phi_A = \max_i (\phi_i), \ i = 4, 5, 6, 7 \).

It might be noticed that in determining \( \min_i \theta_i \) (or \( \max_i \phi_i \)) we considered only four bounding half-planes, that is 4, 5, 6 and 7. The redundant half-plane 8 had intentionally been excluded for if it is included in determining \( \min_i \theta_i \) (or \( \max_i \phi_i \), \( i = 4, 5, \ldots, 8 \) then, as could be seen from figure 3.1, the minimum value now occurs for the half-plane 8, instead of 4. Obviously we would like to guard ourselves against moving to a completely redundant half-plane and hence do not follow this latter criterion in selecting the bounding hyperplanes for our moves in different iterations. (One salient feature, however, emerges from the above discussion, namely that if a half-plane satisfies both (3.5.15) and (3.5.16) then it must pass through the optimum point M).
If we, therefore, follow the criterion given by (3.5.15) then it adequately takes care of excluding the (completely) redundant half-planes and also ensures a move in the relevant feasible and/or infeasible region in our search for the optimum point. It may also be observed here that to reach the half-plane 4, we have to move (whichever axis we might choose) into the infeasible region. This, however, need not necessarily be the case in all situations. For example if we remove the half-planes numbers 2 and 5 from consideration for a moment (their removal, as we can see from the figure, does not effect the optimum point M) then the point R on the $x_2$-axis to which we move is in the feasible region; moving on $x_1$-axis (for reaching the half-plane 4) would, however, still lead to an infeasible point P, say, not shown in the figure. The decision whether to move to the point R or P is guided by consideration of the maximal increase in the optimal function (and not by consideration of movements in the preselected feasible or infeasible region) so that we would move to the (invisible) point P in the above example. The movement to a feasible or an infeasible point will, in each iteration, however, depend on (i) the constraint system and (ii) the functional of the given l.p. problem. The latter perhaps needs some elaboration for clearer understanding. Let us consider that in our example, the optimal function (3.5.1) is changed to,

21. This, in general, would include any such redundant bounding half-plane as number 7 in the infeasible region of figure.
\[ F = -c_1 x_1 + c_2 x_2 \]  
\[ (3.5.17) \]

instead, so that the functional \( Z \) now is

\[ Z = c_1 x_1 - c_2 x_2 \]  
\[ (3.5.18) \]

The selected half-plane would then in this situation be one that is nearest to the origin 0 in the increasing direction of the (mod) \( Z = c_2 x_2 \) plane. It may be noted here that we do not thus (in the first iteration) consider the term ' \(-c_1 x_1\)' of the optimal function (3.5.17) or in other words the term ' \(+c_1 x_1\)' of the functional (3.5.18). This is because the maximum increase (from the origin 0) can, in this case, obviously be obtained by moving along the \( x_2 \)-axis alone. Moreover, having thus selected the nearest bounding half-plane, we would, in this case, move to it also by the \( x_2 \)-axis alone (in general, it would be one of the mod \( Z_j (j \in J) \) axes). This is because we restrict our moves in the sub-space \( A \).

To summarise, we move in each iteration of the B.H.P. method from one vertex to another in the sub-space \( A \) defined by the set of those non-basic variables for which \( Z_j < 0, j \in J \).

The decision whether the vertex, that we move to, lies in the feasible or the infeasible region is governed entirely by the sub-space \( A \). The movements on the bounding hyperplanes in the
increasing direction of the normal to the mod $Z_j$ plane (hyper-plane, in general) keeps us directing towards the optimum point in each iteration till we finally reach the latter, if it exists. And as the initial starting point could be either feasible or infeasible we need also not concern ourselves with the problem of starting the moves always from an initial basic feasible solution point (as in the Simplex Method).

3.6 Estimation of the running time.

Excluding the bookkeeping operations, there will approximately be $\frac{1}{2} (5mn + 6n + 5n)$ multiplications and $\frac{1}{2} mn (5n - 1)$ additions in each iteration. If $\nu$ and $\gamma$ denote the multiply and add times of the Computer respectively then the time for an iteration is about

$$\frac{1}{2} \left\{ (5mn + 6n + 5n) \nu + mn (5n - 1) \gamma \right\} \tag{3.6.1}$$

and if $\Omega$ may denote the number of iterations required to reach the optimal solution then the total time required will be of the order of

$$\frac{\Omega}{2} \left\{ mn (5\nu + 3\gamma) + mn (6\nu - \gamma) + 5mn \gamma \right\} \tag{3.6.2}$$

3.7 Illustrations.

We next illustrate the method with examples.
3.7.1 Example 1: Consider the problem of maximising

\[ F = x_1 + 1.1x_2 \]

subject to the conditions,

\[
\begin{align*}
2x_1 + x_2 & \geq 4 \\
2x_1 + 3x_2 & \geq 6 \\
x_1 - 2x_2 & \leq 4 \\
x_1 + 2x_2 & \geq 6 \\
x_1 + 4x_2 & \geq 8 \\
x_1 - x_2 & \leq 8 \\
5x_1 - 3x_2 & \leq 50 \\
4x_1 - x_2 & \leq 48 \\
5x_1 + x_2 & \leq 75 \\
-4x_1 + x_2 & \leq 1.5 \\
-3x_1 + x_2 & \leq 4 \\
-2x_1 + x_2 & \leq 5 \\
x_1 + x_2 & \leq 6 \\
-2x_1 + 3x_2 & \leq 21 \\
-x_1 + 3x_2 & \leq 27
\end{align*}
\]
\[ \begin{align*}
  x_1 + 12x_2 & \leq 168 \\
  3x_1 + 13x_2 & \leq 169 \\
  -x_1 + 4x_2 & \geq 0 \\
  x_1 - 3x_2 & \leq 1 \\
  \text{and} & \quad x_j \geq 0 \quad j = 1,2
\end{align*} \]

Rewriting the above problem in the form given by (3.1.1) - (3.1.3), we obtain,

maximise

\[ x_1 + 1.1x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6 + 0x_7 + 0x_8 + 0x_9 + 0x_{10} + 0x_{11} + 0x_{12} + 0x_{13} + 0x_{14} + 0x_{15} + 0x_{16} + 0x_{17} + 0x_{18} + 0x_{19} + 0x_{20} + 0x_{21} \]

subject to

\[ \begin{align*}
  -2x_1 - x_2 + x_3 & = -4 \\
  -2x_1 - 3x_2 + x_4 & = -6 \\
  x_1 - 2x_2 + x_5 & = 4 \\
  -x_1 - 2x_2 + x_6 & = -6 \\
  -x_1 - 4x_2 + x_7 & = -8 \\
  x_1 - x_2 + x_8 & = 8 \\
  5x_1 - 3x_2 + x_9 & = 50 \\
  4x_1 - x_2 + x_{10} & = 48 \\
  5x_1 + x_2 + x_{11} & = 75 \\
  -4x_1 + 0.5x_2 + x_{12} & = 1.5 \\
  -3x_1 + x_2 + x_{13} & = 4 \\
  -2x_1 + x_2 + x_{14} & = 5 
\end{align*} \]
\[-x_1 + x_2 \quad +x_{15} \quad = 6 \]
\[-2x_1 + 3x_2 \quad +x_{16} \quad = 21 \]
\[-x_1 + 3x_2 \quad +x_{17} \quad = 27 \]
\[x_1 + 12x_2 \quad +x_{18} \quad = 168 \]
\[3x_1 + 13x_2 \quad +x_{19} \quad = 169 \]
\[x_1 - \frac{4}{3}x_2 \quad +x_{20} \quad = 0 \]
\[x_1 - 3x_2 \quad +x_{21} \quad = 1 \]

and \(x_j > 0, \ j = 1, 2, \ldots, 21\).

The initial point is
\[(0, 0, -4, -6, -8, 8, 50, 48, 75, 1, 5, 4, 5, 6, 21, 27, 168, 169, 0, 1)\]

which, it should be noted, is an infeasible (basic) solution.

The value of the objective function at this point is obviously equal to 0. The new point reached in iteration 1 is
\[(56.3, 0.0, 108.7, 106.7, -52.3, 50.3, 48.3, -48.3, -231.7, -177.3, -206.7, -226.8, 173.0, 117.7, 62.3, 133.7, 83.3, 111.7, 0.0, -55.3, -56.3)\]

with the new objective function value equal to 56.33. The point reached is still infeasible; however, in iteration 2, we reach the optimum point,
\[(13, 10, 32, 50, 11, 27, 45, 5, 15, 6, 0, 48.5, 33, 21, 9, 17, 10, 35, 0, 18, 27)\]

which is a feasible point with all \(Z_j > 0\) and the optimum value equal to 24. The calculations in each iteration are shown on the right hand side of the tables concerned with the particular iteration and are self-explanatory. Referring to the 2-dimensional diagram in figure 3.3 on page...
53 we observe that the point reached in iteration 1 is given by \( L \), and that the next move leads us straight to the optimum point \( G \). If, however, we had chosen to move along the (basic) feasible points alone then, in the first place, we would require at least two iterations to get to a feasible point, say, \( B \) in the diagram and subsequently require at least 5 iterations to reach the optimum point. Moreover, we would encounter the degeneracy situation in the point \( C \) and that consumes an additional iteration to get away from the then current feasible point \( C \) to the next feasible point \( D \). We would, therefore, require a total of 8 iterations to reach the optimum point in this case, as against 2 iterations in following the bounding hyperplane method that allows us to move in either the infeasible region alone or the feasible region alone or else both the infeasible and feasible regions, till we reach the optimum point which of course has to be in the feasible region only. In the present example, the moves were all in the infeasible region, till we reached the optimum point \( G \) in the feasible region.

37.2 Example 2: Consider another example of minimising

\[-\frac{3}{4}x_1 + 150x_2 - \frac{1}{50}x_3 + 6x_4\]

subject to the conditions,

\[
\begin{align*}
\frac{1}{4}x_1 - 60x_2 - \frac{1}{25}x_3 + 9x_4 & \leq 0 \\
\frac{1}{2}x_1 - 90x_2 - \frac{3}{50}x_3 + 3x_4 & \leq 0 \\
x_3 & \leq 1
\end{align*}
\]

and \( x_j \geq 0, \ j = 1,2,3,4. \)

\[^{21}\text{The point } L \text{ is the } X'n \text{ of hyperplane no. 17 with } x_4 \text{-axis and is not shown in the figure 3.3.}\]
The above is an unpublished example [35] of cycling in terms of the primal (for the Simplex Method) due to Beale. Since, we have been considering the general linear programming problem as always consisting of a maximising objective function, we convert the minimising function to the maximising one by multiplying the former by \(-1\); rewriting the problem in the form (1.7) to (1.9), we have:

maximise

\[ 0.75x_1 - 150.00x_2 + 0.02x_3 - 6.00x_4 + 0.0x_5 + 0.0x_6 + 0.0x_7 \]

subject to the conditions,

\[
\begin{align*}
0.25x_1 - 60.00x_2 - 0.04x_3 + 9.00x_4 + x_5 &= 0 \\
0.50x_1 - 90.00x_2 - 0.02x_3 + 3.00x_4 + x_6 &= 0 \\
x_3 + x_7 &= 1 \\
\end{align*}
\]

and \(x_j \geq 0, j = 1,2,\ldots,7\).

The calculations are shown in the adjoining tableaux, which are self explanatory.

The initial point is \((0.00,0.00,0.00,0.00,0.00,0.00,1.00)\)

The point remains stationary in the first iteration. In the next iteration, however, the optimum point given by

\[(0.04, 0.00, 1.00, 0.00, 0.03, 0.00, 0.00)\]

is reached with the optimal value of the objective function equal to \(1/20\).

It may be of interest to mention here that the above artificially constructed example to illustrate cycling (with seven bases) in the Simplex Method takes, even after using \(\varepsilon\)-
\[
\begin{align*}
&\text{Calculations: } (\exists \in T)(T > 0)
\end{align*}
\]

| \(z\) | 00.00 | 00.00 | 00.00 | 00.00 | 6.00 | -0.02 | 015.00 | 0.75 | -0.75 |
|---|---|---|---|---|---|---|---|---|
| 00.00 | 00.00 | 00.00 | 00.00 | 00.00 | 1.00 | 00.00 | 00.00 | 0.00 |
| 00.00 | 00.00 | 00.00 | 00.00 | 00.00 | 3.00 | -0.02 | 00.00 | 00.00 |
| 00.00 | 00.00 | 00.00 | 00.00 | 00.00 | 9.00 | -0.04 | 00.00 | 00.00 |
| \(\chi = 0\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) |
| \(\alpha\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) |
| \(\beta\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) |
| \(\gamma\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) | \(d\) |

---

**Table:**
\[ p \approx 0.33 \]

Since \( z = 8 \) from (3.2.5.6),
\[ u = \frac{z}{\gamma} = \frac{8}{20.00} \]
\[ \frac{0.00}{0.00} \]

<table>
<thead>
<tr>
<th>( z )</th>
<th>( f_z(0) )</th>
<th>( f_z(1) )</th>
<th>( f_z(2) )</th>
<th>( f_z(3) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.15</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.30</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.45</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.60</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.75</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

\[ I \]
Note: 1. The figure shown in the bottom extreme right cell gives the value of the objective function.

|      | 0.00 | 0.05 | 1.00 | 1.50 | 2.00 | 2.50 | 3.00 | 3.50 | 4.00 | 4.50 | 5.00 | 5.50 | 6.00 | 6.50 | 7.00 | 7.50 | 8.00 | 8.50 | 9.00 | 9.50 | 10.00 | 10.50 | 11.00 | 11.50 | 12.00 | 12.50 | 13.00 | 13.50 | 14.00 |
|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 1.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 | 0.04 |
| 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 |

Solution point 15 as the condition (5.2.10) is true the optimum point is reached. The optimum value is 0.05 and the solution point is 0.03.
perturbation schemes for resolving degeneracy, six iterations to reach the optimum point [36].

3.7.3 Example 3. Let us consider Beale's [5] example (in terms of the dual also) that illustrates cycling in the Dual Simplex Algorithm [36]. The problem is to

minimise

\[ F = x_3 \]

subject to

\[-\frac{1}{4}x_1 - \frac{1}{2}x_2 + x_4 = -\frac{3}{4} \]

\[ 8x_1 + 12x_2 + x_3 = 20 \]

\[ x_1 + \frac{1}{2}x_2 - x_3 + x_6 = -\frac{1}{2} \]

\[ -9x_1 - 3x_2 + x_7 = 6 \]

and \[ x_j \geq 0, \ j=1,2,\ldots,7. \]

The minimising function could be rewritten as a maximising function in the form

\[ F = -x_3 \]

This example calls for the direct application of class S rules alone. The calculations are shown in the adjoining tableaux. It may be observed that the optimum point given by

\( (0, 3/2, 5/4, 0, 20, 21/2) \)

is reached in two iterations. The problem otherwise cycles with six bases in the Dual Simplex method and can be solved only by following one of the \( \varepsilon \)-perturbation schemes.
\[ p_q = 1 - \frac{1}{z} \]

and \( \frac{1}{z} = \frac{1}{T} \).

\[ \therefore \quad z = \frac{1}{T} \]

\[ \therefore \quad \phi = \frac{1}{T} \]

Since \( x \to 0 \) and \( z \to 0 \), rules of class \( S \) will apply, thus:

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & f_z \\
6 & 1 & 0 & 0 & 0 & 0 & 0 & -3 \\
2/2 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & -1 \\
20 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
-3/4 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & f_z \\
7 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & -1 \\
5 & 4 & 3 & 2 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & f_z \\
\end{array}
\]

The table above shows the calculations for the given conditions.
\[
\text{hence, pivot } \frac{Q_{12}}{Q_{11}} = T_2
\]

From (3.2.15): $\text{Max} (4, -4, 1) \text{ corresponds to } T_2$

\[
T_1 = 3/2, \quad \forall \quad T_1 = 3/2, \quad \text{Rule of Gauss Jordan, etc.}
\]

Calculations

<table>
<thead>
<tr>
<th>$z_2$</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>3</th>
<th>6</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>8</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>$-3/4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>12</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3.5

\text{Iteration 1}
2. The pivotal element is indicated by a circle around it.

1. The figure shown in the bottom extreme right cell gives the value of objective

\[
\begin{align*}
\frac{y}{x} &= \frac{2}{7} \\
\frac{y}{x} &= \frac{3}{2} \\
\frac{y}{x} &= \frac{3}{2} \\
\frac{y}{x} &= \frac{3}{2} \\
\frac{y}{x} &= \frac{3}{2}
\end{align*}
\]

Note: 1. The figure shown in the bottom extreme right cell gives the value of objective

The optimal value is \( \frac{3}{4} \) and the solution point is \( \frac{3}{4} \). As the condition \( \frac{3}{4} \leq 1 \) is true, the optimum point is reached.

Calculation

<table>
<thead>
<tr>
<th>( \frac{y}{x} )</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>( \frac{y}{x} )</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/12</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2/12</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5/4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3/4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Iteration 2

Table 3.9
4.1 Equalities and their relationship to inequalities.

As stated in the last chapter, we now consider the case when the equalities occur together with the inequalities. In this case it is essential that the former must be satisfied exactly in the optimum point. We, therefore, solve first the equality constraints independently of the inequality constraints and if the 'equality' solution tableau so obtained does not yield an inconsistent solution point then the former is utilised to obtain the 'transformed' inequality tableau (corresponding to the inequality constraints) for the current 'equality' solution point. The set of the transformed constraints thus obtained, corresponding to both the equality and inequality constraints, is now a l.p. problem with only inequality constraints and is solved by the application of the B.H.P. method discussed in chapter 3.

Let us reconsider the l.p. problem formulated in section 2.1, chapter 2. We, for the sake of convenience of description, however, slightly change the notation so that the column vector \( P_c \) is denoted by \( P_A \), and its elements which may be negative and/or non-negative are given by
Let us suppose the augmented matrix $(A | P_A) = \begin{bmatrix} a_{1,n+1} \\ a_{2,n+1} \\ \vdots \\ a_{m,n+1} \end{bmatrix}$ to be partitioned into two submatrices

$(V | P_V) = \begin{bmatrix} \beta_{1,j} \\ \beta_{2,j} \\ \vdots \\ \beta_{n,j} \end{bmatrix}$ an $m_1$ by $n+1$ matrix and

$(W | P_W) = \begin{bmatrix} \alpha_{1,j} \\ \alpha_{2,j} \\ \vdots \\ \alpha_{n',j} \end{bmatrix}$ an $m_2$ by $n+1$ matrix corresponding to the inequality and the equality constraints respectively where

\[ m = m_1 + m_2 \tag{4.1.2} \]

Let $I$ and $J$ be the sets of indices of the rows and columns respectively of $A = (V | W)$ and let $I_V$ and $I_W$ be the sets of indices of rows of $V$ and $W$ respectively so that

\[ I_V \cup I_W = I ; \quad I_V, I_W \subseteq I \tag{4.1.3} \]

Further, if the submatrix $W$ comprises of the null column vectors, if any, numbering say $n'$ then let $J' \subset J \setminus J_{k\alpha} \subset J$ define the sets of indices of columns of $W$ that are/are not the null column vectors so that
\[ J' \cup J_{\text{eq}} = J \quad \text{(4.1.4)} \]

and

\[ \tau = n - n' \quad \text{(4.1.5)} \]

where \( \tau \) defines the effective number of columns of \( W \).

4.2 The equality constraints. As stated above we consider first the submatrix \( W \) independently, for the time being, of the submatrix \( V \). Since the equality constraints must be satisfied exactly we make the condition that an equality once selected as a pivotal row will not be reconsidered for pivotal row selection in the subsequent iterations. For this purpose, we let

\[ I_{\bar{W}}^e \subseteq I_W \]

denote the set of indices \( i \) for which the equality hyperplanes \( i (\in I_W) \) have been considered and let \( I_{\bar{W}}^{ce} \subseteq I_W \) denote the set of indices \( i \) for which the hyperplane \( i (\in I_W) \) are yet to be considered so that

\[ I_{\bar{W}}^e \cup I_{\bar{W}}^{ce} = I_W \quad \text{(4.2.1)} \]

where \( \bar{W} = (\bar{\alpha}_{ij}) \) denotes the transformed matrix \( W \) at the points reached in different iterations. (The latter may significantly vary depending upon the technique deployed in the solution of the equality constraints for, with the possibility of starting with an infeasible solution point as well in the B.H.P. method we are at liberty to choose any one out of a number of
techniques in the literature). Further if the row vector
\( Z (= Z_j , j \in J) \) denotes the transformation to the functional 
'\( -CX \)' in different iterations then let \( J_1 \subseteq J_{eq} \) denote
the set of indices \( j \) for which

\[
Z_j < 0 \quad , j \in J_1 \quad , \quad (4.2.2)
\]

and also if \( \overline{W}_i , i \in I_{\overline{W}} \) defines the row space of \( \overline{W} \) then
let \( \overline{W}_i^j \ (i \in I_{\overline{W}} , j \in J_1) \) denote the corresponding
row elements of \( \overline{W}_i \) for which \((4.2.2)\) holds.

### 4.2.1. Techniques.

We now describe two techniques for the solution of equality
constraints. Let \( \overline{W}_{\eta \gamma} \) in both the techniques denote, in
general, the pivotal element where \( \eta \), as before, stands for
the pivotal row and \( \gamma \) the pivotal column.

1. **Technique 1.** The pivotal row \( \eta \ (= i \in I_{\overline{W}}) \) in this
approach is chosen by successively considering each of the \( m_2 \)
rows (starting from the first) in that order. Then if

\[
\left( \forall j \in J_{eq} \right) \left( Z_j > 0 \right) \quad (4.2.3)
\]

\( \gamma \) is chosen from

\[
\max j \left| \overline{C}_{\eta_j} \right| \quad , j \in J_{eq} \quad , \quad (4.2.4)
\]

but if

\[
\left( \exists j \in J_{eq} \right) \left( Z_j < 0 \right) \quad (4.2.5)
\]
then $\gamma$ is chosen from

$$\max_{j} |\overline{W}_{\gamma}^{j}|, \; j \in J_{eq},$$  \hspace{1cm} (4.2.6)

if it exists, otherwise it is selected from (4.2.4).

The new tableau is obtained by carrying out the pivotal transformations using formulae (4.2.21). The equality operations terminate as discussed in section 4.2.3.

(ii) Technique 2. In this approach we utilise the concepts developed for the B.H.P. method in chapter 3. We first distinguish, as in the technique 1 above, the two situations given by (4.2.5) and (4.2.3). It may be recalled that these two situations are termed the Classes R and S in the last chapter; refer (3.2.1) and (3.2.11). To maintain analogy we rename them as classes RE and SE (abbreviation for equality). The pivotal element $\overline{\alpha}_{\eta}^{\gamma}$ is then determined following the rules discussed below for the two classes.

Class RE. $((3j \in J_{eq}) (Z_{j} < 0))$  \hspace{1cm} (4.2.7)

Rule RE.1. Select the pivotal row $\eta (= i \in \overline{I_{eq}})$ as that hyperplane which is farthest in the increasing direction of the normal to the plane from the origin. It is given by

$$\max_{i} \left| \frac{\overline{\alpha}_{i,n+1} / \sum_{j} \overline{\alpha}_{i,j} Z_{j}}{Z_{j}} \right|, \; i \in \overline{I_{eq}}, \; j \in J_{eq}. \hspace{1cm} (4.2.8)$$
If a tie occurs then we select any of the \( i \in I_{w}^{eb} \) for which the tie occurs.
This is termed the exit criterion for the class \( R \).

Next we examine if

\[
(i) \quad \bar{\alpha}_{\eta, n+1} > 0 , \quad (4.2.9)
\]

or

\[
(ii) \quad \bar{\alpha}_{\eta, n+1} < 0 . \quad (4.2.10)
\]

Rule RE.2. The entry criteria for the two cases above are as below.

**RE.2.1.** If \((4.2.9)\) is true then \( \gamma (= \min J_{EA}) \)
is chosen from

\[
\min_{j} \left( \frac{Z_{j}}{\bar{\alpha}_{\eta_{j}}} \right), \bar{\alpha}_{\eta_{j}} < 0 , j \in J_{EA} ; \quad (4.2.11)
\]

but if

\[
\left( (\forall j \in J_{EA}) (\bar{\alpha}_{\eta_{j}} > 0) \right) \quad (4.2.12)
\]

so that \( \gamma \) cannot be selected from \((4.2.11)\) then we choose it from

\[
\min_{j} \left( \frac{Z_{j}}{\bar{\alpha}_{\eta_{j}}} \right), \bar{\alpha}_{\eta_{j}} \neq 0 , j \in J_{EA} ; \quad (4.2.13)
\]

**RE.2.2.** If \((4.2.10)\) is true then \( \gamma (= \min J_{EA}) \) is chosen from

\[
\max_{j} \left( \frac{Z_{j}}{\bar{\alpha}_{\eta_{j}}} \right), \bar{\alpha}_{\eta_{j}} > 0 , j \in J_{EA} \quad (4.2.14)
\]

214. If a tie occurs then we select any of the \( j \in J_{EA} \) for which the tie occurs.
but if
\[ ((\forall j \in J_{ER})(\bar{\alpha}_{ij} < 0)) \]  \hspace{1cm} (4.2.15)
then we select \( \gamma \) alternatively from
\[ \max_j (Z_j / \bar{\alpha}_{ij}), \bar{\alpha}_{ij} \neq 0, j \in J_{EA}. \]  \hspace{1cm} (4.2.16)
This is termed the entry criterion for the class RE.

**Class SE**
\[ ((\forall j \in J_{EA})(Z_j \geq 0)) \]  \hspace{1cm} (4.2.17)

**Rule SE.1.** Select the pivotal row \( \eta \) as any one of the hyperplanes \( i \in I_{\omega} \). This is termed the exit criterion for the class SE.

**Rule SE.2.** Select \( \gamma (= j \in J_{EA}) \) from
\[ \min_j |Z_j / \bar{\alpha}_{ij}|, \bar{\alpha}_{ij} < 0, j \in J_{EA}. \]  \hspace{1cm} (4.2.18)
but if
\[ ((\forall j \in J_{EA})(\bar{\alpha}_{ij} \geq 0)) \]  \hspace{1cm} (4.2.19)
then \( \gamma \) is chosen from
\[ \min_j (Z_j / \bar{\alpha}_{ij}), \bar{\alpha}_{ij} \neq 0, j \in J_{EA}. \]  \hspace{1cm} (4.2.20)
(A tie is handled as in the previous case.) This is termed the entry criterion for the class SE.

**4.2.2. The new tableau.** This is obtained at the end of each iteration by performing the Gaussian eliminational transformations on the pivotal element \( \bar{\alpha}_{\eta \gamma} \), from
\[
\begin{align*}
\bar{\alpha}'_{ij} &= \bar{\alpha}_{ij} - \frac{\bar{\alpha}_{\eta j}}{\bar{\alpha}_{\eta \gamma}}, i \neq \eta) \in I_{W+1}^+ \\
\bar{\alpha}'_{i\eta} &= \frac{\bar{\alpha}_{i\eta}}{\bar{\alpha}_{\eta \gamma}}, i = \eta
\end{align*}
\]

(4.2.21)

where a prime denotes the new values of the elements in the new tableau. The new values of the optimal function \( F \), the functional \( Z \) and the solution vector \( \bar{\alpha}_{i, n+1} \), \( i \in I_{W} (= \bar{X}_{w}) \) respectively are given by

\[
\begin{align*}
\bar{\alpha}'_{\eta i} &= \bar{\alpha}_{m_{2}+1, n+1} = F \\
\bar{\alpha}'_{\eta j} &= \bar{\alpha}_{m_{2}+1, j}, j \in J_{\text{eq}} = Z (= Z_{j}), \\
\bar{\alpha}'_{i\eta} &= \bar{\alpha}_{i, n+1}, i \in I_{W} = \bar{X}_{w} (= \bar{a}_{i, n+1})
\end{align*}
\]

4.2.3. The final equality \((\bar{W} \mid P_{\bar{W}})\) tableau.

If \( I_{\bar{W}}^{ce} = I_{\bar{W}}^{red} \), that is,

\[
I_{\bar{W}}^{e} \cup I_{\bar{W}}^{red} = I_{\bar{W}} \quad (4.2.23)
\]

where \( I_{\bar{W}}^{red} \subset I_{\bar{W}}^{ce} \) denotes the set of indices \( i \) (redundant hyperplanes) for which

\[
(( \exists i \in I_{\bar{W}}^{ce}) ((\bar{\alpha}_{i, n+1} = 0) \& (\forall j \in J_{\text{eq}})(\alpha_{ij} = 0)))
\]

(4.2.24)

then the equality operations of the section 4.2 terminate. The
Tableau so obtained is termed the final equality tableau. If in this tableau it is found that

\[
\left( \exists i \in I_{\overline{W}} \right) \left( \exists \alpha_{i, n+1} \neq 0 \right) \land \left( \forall j \in J_{\alpha} \right) \left( \alpha_{i,j} = 0 \right)
\]

then it implies that the solution to the problem does not exist (inconsistent solution point).

If (4.2.25) is not true and

\[
\gamma_1 = 0
\]

then the optimal solution, if it exists, is obtained as discussed in section 4.4, but if

\[
\gamma_1 \neq 0
\]

then we develop in section 4.3 the relations for obtaining the transformations to the inequality constraints \( (V|P_V) \) from the final equality tableau.

4.2.3.1. Motivation in selecting one of the two techniques (i) and (ii).

It may be useful to mention here the reasons that prompted us for suggesting the two techniques given above. The technique 1 takes care of reducing the rounding errors by selecting the pivot as that element which has the largest magnitude in the pivotal row. This is strictly the case in the situation (4.2.3).

There is, however, a compromise attempted with situation (4.2.5) when the pivotal element is the one which has the largest magnitude corresponding to negative coefficients of the functional (refer (4.2.2)); this is because we know that the (maximum) increase to the optimal function will occur by selecting a variable from
Table 4.1.

<table>
<thead>
<tr>
<th>V (β_{i,j} an ( m_2 \times n ) submatrix)</th>
<th>( P_V (\beta_{i,n+1}^{m_2}) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>A (a_{i,j} an ( m \times n ) matrix)</td>
<td>( P_A (\alpha_{i,n+1}^{m_1}) )</td>
</tr>
<tr>
<td>W (\alpha_{i,j} an ( m_2 \times r ) submatrix)</td>
<td>( P_W (\alpha_{i,m+1}^{m_2}) )</td>
</tr>
</tbody>
</table>

Null Matrix (=m_2xn)

Functional

\( J \)

Organisation of the augmented matrix \((A | P_A)\) and the functional.
the set \( J_1 \) defined by (4.2.2). In the technique 2, on the other hand, we have essentially utilised the l.p. concepts developed for the B.H.P.M. in chapter 3. If the equality constraints are not inconsistent we would usually result at the end of equality operations of technique 2 in the situation (4.2.17) — refer class SE, and this is a considerable advantage in reducing the number of iterations of the problem. (In some cases the solution point obtained at the end of technique 2 may well be the optimum point for the (complete) l.p. problem). If we end up with the class RE at the end of technique 2 then we have obtained the maximum increase in the optimal function or else the minimum decrease (if that be the case) in each iteration and this has been directly related to the (real) optimal function (2.1.1) and not to an auxiliary function as in the Simplex Method.

4.3 The final inequality/functional tableau.

Let the constraints (2.1.2) and the functional \( Z \) be represented as shown in table 4.1. Also let the final equality matrix (tableau) be as shown in the middle portion of table 4.2. The transformations to the function \( F \) and the coefficients \( \beta_{ij} \) of the final inequality tableau are then obtained, using the coefficients \( \alpha_{ij} \) of the final equality tableau, from

Table 4.2

<table>
<thead>
<tr>
<th>( \bar{\beta}_{ij} )</th>
<th>( \bar{\beta}_{i1} = 0 )</th>
<th>( \bar{\beta}_{i1} )</th>
<th>( \bar{\beta}_{i,n+1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Null Matrix</td>
<td>( Z_1 = Z_1 )</td>
<td>( Z_1 = 0 )</td>
<td>( Z_4 )</td>
</tr>
</tbody>
</table>

Final equality (after phase 1 of the Simplex Method) and the Inequality/Functional tableau.
\[
\overline{\beta}_{ij} = \beta_{ij} - \sum_{k=1}^{n_1} \beta_{ik} g_k \bar{\alpha}_{k,ij}, \quad i \in I_V, j \in J, \quad (4.3.1)
\]

\[
\overline{\beta}_{i,n+1} = \beta_{i,n+1} - \sum_{k=1}^{n_1} \beta_{ik} g_k \bar{\alpha}_{k,n+1}, \quad i \in I_V; \quad (4.3.2)
\]

where

\[
G = \{ g_i \} \quad 1 \leq i \leq n_1, \quad (4.3.3)
\]

\[
H = \{ h_i \} \quad 1 \leq i \leq n_1.
\]

\(n_1 (\leq r)\) is the column rank of \(W\) and \(g_i\) and \(h_i\),

\(i = 1, 2, \ldots, n_1\) are the elements of the index vectors \(G\)

and \(H\) that record the column and row respectively of the

successive elements of the \(W\) matrix used as pivots at each

stage of the operations of section 4.2.1. The coefficients \(\overline{\beta}_{ij}\) (for the inequality part of the table 4.2) computed from

\((4.3.1)\) relate, however, to only those columns in the final

equality tableau that correspond to the column nullity

\(r - n_1 (n_1 < r)\) in the original submatrix \(W\). This is

because the coefficients in the I/F tableau corresponding to the

null matrix, if any, in the \(W\) submatrix (or equality tableau)

remain unchanged as the terms

\[
\sum_{k=1}^{n_1} \beta_{ik} g_k \bar{\alpha}_{k,j} = 0, \quad i \in I_V, j \in J'; \quad (4.3.4)
\]

on the right hand side of \((4.3.1)\) hence,
\[
\beta_{ij} = \beta_{ij}, \quad j \in J', i \in I_v
\]  

(4.3.5)

And the coefficients in the I/F tableau corresponding to the unit matrix coefficients in the final equality tableau are necessarily zero (that is a null matrix).

We next consider if, in the I/F tableau,

\[
((\exists i \in I)((\beta_{i,n+1} < 0) \& (\forall j \in J)(\beta_{ij} \geq 0)))
\]  

(4.3.6)

which implies that the optimal solution to the problem does not exist: If (4.3.6) is not true then we concern ourselves with relevant parts of the direction analysis discussed generally below.

4.4 Directional analysis. As mentioned earlier, we distinguish two classes of situation in the given linear programming problem,

Class a: \( m_2 \neq 0, m_1 = 0 \) (equalities only);

Class b: \( m_2 \neq 0, m_1 \neq 0 \) (both equality and inequality constraints).

The following cases may exist in both the classes:

\[
\begin{aligned}
(i) & \quad m_2 > \tau \\
(ii) & \quad m_2 = \tau \\
\text{and (iii) } & \quad m_2 < \tau
\end{aligned}
\]  

(4.4.1)
We consider the three cases in each class separately,

**Class a.** (i) If it is found that \( d (= m_2 - \tau) \) hyperplanes are redundant (that is the number of unit column vectors is exactly equal to \( \tau \)) then a unique solution to the problem exists; proceed to 4.4.1. However if
\[
d > m_2 - \tau \quad , \quad (4.4.2)
\]
then an infinite number of solutions exist in which case proceed to 4.4.2.

(ii) If there is/are no redundant hyperplanes then a unique solution to the problem exists; proceed to 4.4.1. However, if there is at least one redundant hyperplane (that is the number of unit column vectors is less than \( \tau \)) then an infinite number of solutions exist; proceed to 4.4.2.

(iii) An infinite number of solutions exist; proceed to 4.4.2.

**Class b:**

Let \( J_2 \subset J_{\mathbb{R}} \) denote the set of indices that correspond to those columns (or variables) of the \( W \) submatrix that have column nullity \( \tau - n_\nu \) and let \( J'' \) define the set given by
\[
J'' = J' \cup J_2 \quad . \quad (4.4.3)
\]
Then if
\[
(( \forall i \in I_V)(\forall j \in J'')(\bar{b}_{ij} = 0)) \quad (4.4.4)
\]
the considerations in class a apply exactly to
this situation and we follow the directions of the three cases as discussed therein. If (4.4.4) is not true then there exists an infinite number of solutions and we proceed to 4.4.2.

4.4.1 The unique solution: There is an optimal solution to the problem only if

\[(\forall i \in I) (a_{i,n+1} \geq 0) \& (\forall j \in J) (L_j \geq 0)\]  

(4.4.5)

where

\[\bar{a}_{i,n+1} = \bar{b}_{i,n+1}, \quad i \in I \bigcup \bigcup\]  

(4.4.6)

\[\bar{a}_{i,n+1} = \bar{b}_{i,n+1}, \quad i \in I \bigcap \bigcup\]  

(4.4.7)

otherwise the optimal solution does not exist.

4.4.2 An infinite number of solutions. These are categorised in the following three mutually exclusive and collectively exhaustive cases which uniquely direct the next move:

(i) if (4.4.5) is true,

then the current solution point given by

\[\bar{x}_{i,n+1} = \bar{a}_{i,n+1}, \quad i \in I\]  

(4.4.8)

is the optimal point, where \(\bar{a}_{i,n+1}\) are given by (4.4.6) - (4.4.7).

(ii) if \(\left(\forall j \in J \right) (L_j \geq 0) \& \left(\exists i \in I \right) (a_{i,n+1} < 0)\),

then apply the class S decision rules of the B.H.P. Method chapter 3 or the Dual Simplex Method [56] to obtain the optimal solution.
(iii) If \( \left( \bigwedge_{j \in J} (z_j < 0) \right) \) \hspace{1cm} (4.4.10)

then apply the class R decision rules of the B.H.P. Method, ch.3 to obtain the optimal solution of the problem.

4.5 Estimation of running time.

In view of initially restricting the pivotal transformations to only the equality constraints and then developing the transformed inequality constraints from the former we have been able to effectively reduce the time taken in each iteration in the 'equality operations'. An estimate of the running time for the equality operations alone will be

\[
\frac{1}{2} m'_2 \left[ 3\nu \left( m_2+1 \right) - 3m_2(m_2-1) + 2 \right] + \gamma \left[ 3n(1+3m_2) + m_2(6-5m_2) - 1 \right] \] \hspace{1cm} (4.5.1)

This estimate for the solution of a l.p. problem that comprises both the equality and inequality constraints but does not consider the former independently of the latter, is given by

\[
\frac{1}{6} m'_2 \left[ 3\nu \left( m_2+1 \right) - 3m_2(m_2-1) + 2m_2(2n-m_2+1) + 2 \right] + \gamma \left[ 3n(1+3m_2) + m_2(6-5m_2) + 3m_2(2n-m_2+1) - 1 \right] + \frac{\Omega}{2} \left[ (6n-m_2) \left( 5 \nu + 3 \nu \right) + m(63) \right] m(63) \nu \nu \nu \] \hspace{1cm} (4.5.2)

where \( \Omega, \nu, \gamma \) are as defined in chapter 3. If, however, we followed the approach discussed in this chapter then (4.5.2) reduces to

\[
\frac{1}{6} m'_2 \left[ 3\nu \left( m_2+1 \right) - 3m_2(m_2-1) + 2 \right] + \gamma \left[ 3n(1+3m_2) + m_2(6-5m_2) - 1 \right] + \frac{\Omega}{2} \left[ (n-m_2) \left( 5 \nu + 3 \nu \right) + m(63) \nu \nu \nu \right] \approx \text{approximately} \] \hspace{1cm} (4.5.3)

and an estimated saving in computer running time to the order of
\frac{1}{2} m_1 m_2 (2n-m_3+4) (2v+y) \quad (4.5.4)

is achieved.

4.6 Illustration. We illustrate the method using technique 2 in example 1 below.

Example 1.

Consider the problem of maximising,

\[ F = x_1 + x_2 + x_3 \]

subject to the conditions,

\[ \begin{align*}
5x_1 + 2x_2 + 5x_3 & \leq 10 \\
3x_1 + 3x_2 + x_3 & = 3 \\
2x_1 + 2x_2 + 3x_3 & = 6
\end{align*} \]

The matrix \( \begin{pmatrix} A & P_A \\ V & P_V \\ W & P_W \end{pmatrix} \) is shown in table 4.3.

The subsequent iterations tableau are shown in tables 4.4 to 4.6. We may observe that using technique (ii) for the solution of the equality constraints independently of the inequality constraints in the problem we obtain in the first iteration (tableau 4.4) the class SE situation (4.2.17) where,

\[ Z_j \geq 0 \quad , \quad j \in J_{EQ} \quad (= J, \text{in this case}). \]

We retain this situation by considering the class SE (decision) rules now and obtain the new solution point as given in tableau 4.5. Next using the formulae (4.3.1)-(4.3.2) we obtain the I/F
Table 4.3

<table>
<thead>
<tr>
<th>j</th>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Calculations/Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(i)</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>[ \frac{z_{1e}}{x_{21} x_{31}} ], refer (4.2.8).</td>
</tr>
<tr>
<td>2</td>
<td>(i)</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3/7 [ (4.2.6) \text{ and } (4.2.13) ] \eta = 3 and \gamma = 1; hence the pivotal element is ( x_{31} ).</td>
</tr>
<tr>
<td>3</td>
<td>(i)</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>6/7 [ (4.2.13) ] \eta = 3 and \gamma = 1; hence the pivotal element is ( x_{31} ).</td>
</tr>
<tr>
<td>Z_j</td>
<td></td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.4

<table>
<thead>
<tr>
<th>j</th>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Calculations/Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(i)</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>[ \frac{z_{1e}}{x_{21} x_{31}} ], refer (4.2.8).</td>
</tr>
<tr>
<td>2</td>
<td>(i)</td>
<td>0</td>
<td>0</td>
<td>-7/2</td>
<td>-6 clearly \eta , in this case, is for ( i = 2 ); using (4.2.18) \gamma = 3; hence the pivotal element is ( x_{23} ).</td>
</tr>
<tr>
<td>3</td>
<td>(i)</td>
<td>1</td>
<td>1</td>
<td>3/2</td>
<td>3</td>
</tr>
<tr>
<td>Z_j</td>
<td></td>
<td>0</td>
<td>0</td>
<td>1/2</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4.5

<table>
<thead>
<tr>
<th>j</th>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(i)</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>10 \text{ All the equalities are, at this stage satisfied; the transformation to inequality constraints, using formulae } (4.3.1)-(4.3.4) \text{ are shown in tableau 4.6.}</td>
</tr>
<tr>
<td>2</td>
<td>(i)</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>12/7</td>
</tr>
<tr>
<td>3</td>
<td>(i)</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>3/7</td>
</tr>
<tr>
<td>Z_j</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13/7</td>
</tr>
</tbody>
</table>

Table 4.6

<table>
<thead>
<tr>
<th>j</th>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>P_6(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Basis</td>
<td>( P_1 )</td>
<td>( P_2 )</td>
<td>( P_3 )</td>
<td>( P_4 )</td>
<td>( P_6(%) )</td>
</tr>
<tr>
<td>1</td>
<td>( P_4 )</td>
<td>0</td>
<td>-3</td>
<td>0</td>
<td>1</td>
<td>-5/7</td>
</tr>
<tr>
<td>2</td>
<td>( P_3 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>12/7</td>
</tr>
<tr>
<td>3</td>
<td>( P_4 )</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3/7</td>
</tr>
<tr>
<td>Z_j</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13/7</td>
</tr>
</tbody>
</table>

The B.H.P. tableau at the end of equality operations.
tableau (which is the initial tableau for BHPM as described in chapter 3). This tableau is given in Table 4.6 and as we can see we have, by using the equality technique (ii), obtained the point of class S (chapter 3) alone. The optimum tableau is obtained (not shown) in the next iteration.
Chapter 5

FLOW DIAGRAMS AND THE GENERAL ALGORITHM

5.1 Flow diagrams. The flow diagram given below illustrates all three possibilities; that is when the l.p. problem consists of either inequalities alone (chapter 3) or equalities alone (chapter 4) or both the equalities and inequalities (chapter 4). These are respectively indicated by the value of the trigger 'NTRIG' equal to 'zero', 'one' or 'two'. M and N in the flow diagram below denote the 'number of equations+1' and 'number of variables (without addition of slack variables)+1' in the l.p. problem.

```
start
read values of the parameters M, N and trigger NTRIG.

is NTRIG=0?
   yes
   initialise arrays and matrices by considering M and the total number of variables equal to (N+M)
   read data of given l.p. problem
   convert type II inequalities to type I and add (positive) slack variables
   Q
   is it an optimal solution?
      yes
      print values of optimal function, solution pt and shadow prices
      stop
      optimal solution does not exist
      L
      is problem non-feasible or unbounded?
         yes
         distinguish two classes R & S and choose the pivotal element accordingly
         perform the pivotal operations
         P
         no
         is problem non-feasible or unbounded?
            yes
            print values of optimal function, solution pt and shadow prices
            stop
            optimal solution does not exist
            L
            no
            is it an optimal solution?
               yes
               print values of optimal function, solution pt and shadow prices
               stop
               optimal solution does not exist
               L
               no
               Q
   no
   S
```
is N\text{TRIG} = 1? \\

- yes: initialise arrays and matrices for values of M and N as read \\
- no: read data of given l.p. problem \\
  
  1. treat equalities independently of inequalities, 
  2. count \( n_q \) of equalities and set \( m_2 \) equal to this count, 
  3. initialise arrays and matrices by considering M and the total \( n_q \) of variables equal to \( N+M-m_2 \). 
  
  are all equalities satisfied? \\
  - yes: is N\text{TRIG} = 1? \\
  - no: obtain transformations to inequality constraints \\
    
    - yes: go to Q 
    - no: go to L 
  
  distinguish two classes in either of techniques 1 and 2 whichever we decide to choose and determine pivotal element accordingly 
  
  perform the pivotal transformations
If $P > 0$, print optimal value, solution, and shadow prices.

If $P < 0$, test $M = 0$.

If $P = 0$, set $P = T$ and go to 4.

If not, set $P = T$, test $M = 0$.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.

If $P > 0$, go to 5.

If $P < 0$, go to 4.

If $P = 0$, go to 4.
testing purposes as required subsequent to the algorithm. This and other minor details are not shown in the algorithm above.

The number of equations (w) is computed at the time of reading the data of the given 1.0 problem and is stored separately for later usage. If w is set to 0, 1 or 2 (as the case may be) by reading a parameter card that has the appropriate value of the required in the.

otherwise, the program transcriptions are given by Eqs. (3.21)-(3.2.18). Go to A.2.1.

A2.2.1, go to A.1.2.1.

A2.3. Otherwise, try to find I. From Eqs. (3.2.9) or (3.2.7)/(3.2.9) as the case may be.

A2.4. Otherwise, find I = 1, (3.4.1) - exact criterion class.

A2.5. Otherwise, find I = 1, (3.4.12)/(3.2.18) - exact criterion class.

A2.6. If I = 1, proceed as outlined above.

A2.7. If not, test 0 < 4.

A2.8. If test is false, the program terminates.

A2.9. If test is true, the program transcriptions are given by Eqs. (3.2.1)-(3.2.18). Go to A.2.1.
Chapter 6

APPLICATION OF THE B.H.P. TO THE SIMPLEX METHOD OF LINEAR PROGRAMMING

6.1 The two phases. Let us consider the l.p. problem as formulated in section 2.1 of chapter 2. As we know, the Simplex method works in two phases. Phase 1 determines a basic feasible solution (satisfying the equality constraints) and phase 2 then leads from the solution so obtained to the optimal solution. In considering the application of the B.H.P. to the Simplex method we will, as explained in chapter 4, treat first the equalities independently of the inequalities. For this purpose we consider the application of either (a) one of the two techniques of chapter 4 or alternatively (b) the artificial variable technique of the Simplex method. If the latter were to be utilised then, as stated, we restrict its application to the equality constraints alone. The auxiliary function, as in the Simplex method, is constructed and the usual Simplex rules for the entry and exit criterion are followed. At the end of the amended phase 1 as above we obtain a feasible solution point satisfying the equality constraints. The transformations carried out on the inequality constraints and the functional using the formulae (4.3.1)-(4.3.2) of chapter 4 and

$$\bar{Z}_j = Z_j - \sum_{k=1}^{n} Z_{q_k} \bar{C}_{h_k, j}, \quad j \in J$$  \hspace{1cm} (6.1.1)$$

$$\bar{F} = F - \sum_{k=1}^{n} Z_{q_k} \bar{C}_{h_k, n+1}$$
respectively yield us the enlarged linear programming problem with only inequality constraints and in which all the equalities are satisfied. This problem is in the same format as envisaged in the I/F tableau obtained by considering the techniques in (a) above (refer chapter 4).

The solution point for the enlarged problem may, however, not necessarily be a feasible one; in fact, we would for generality presume that it is an infeasible point. We next proceed (as discussed in chapter 3 and 4) to convert the type I ' \( \geq \) ' type constraints to type II ' \( \leq \) ' constraints and then add a positive slack variable to each constraint so as to obtain the B.H.P. tableau of (3.1.12). We are now ready to institute the extended algorithm for the Simplex method as explained in the following section.

6.2 The extended algorithm. class R; \((\exists j \in J) (Z_j < 0)\).

6.2.1 Phase 0. It pertains to decision rules for the class R. It is termed phase 0 so as to distinguish it from phases 1 and 2 of the Simplex method.

6.2.1.1 Criterion for a variable to leave the basis. This corresponds to the selection of the \( \eta \) hyperplane \((=i \in I)\) according to (3.1.21), chapter 3, class R.

6.2.1.2 Criterion for a variable to enter the basis. This corresponds to the selection of the axis \( \gamma \) \((=j \in J)\) according to either the rule R2.1, (3.2.6) or the rule R2.2, (3.2.7)/(3.2.9) of chapter 3, as the case may be.

6.2.2 New tableau and the evaluations. Perform the usual Gaussian
eliminational transformations, using formulae (3.2.17)-(3.2.18) to obtain the new tableau. We then evaluate the sign conditions of $Z_j, j \in J$ and $x_i, i \in I$; the following four mutually and collectively exhaustive cases exist which direct the next move in the algorithm. These are

(i) if $((\forall j \in J) (Z_j \geq 0) \& (\forall i \in I) (x_i \geq 0))$

then the optimal solution has been found;

(ii) if $((\exists j \in J) (Z_j < 0) \& (\exists i \in I) (x_i < 0))$

then, if $\Delta F$ (given by (3.2.3), chapter 3) is non-negative proceed to 6.2.1, phase 0. above;

otherwise proceed to 6.2.3, phase 1 below. The infeasible point now obtained will, in general, yield (on application of phase 1 of the Simplex method) a feasible point which is nearer to optimum point than the feasible point that would have been obtained from the initially available infeasible point. Refer example 1, section 6.3, that illustrates this point.

(iii) if $((\exists j \in J) (Z_j < 0) \& (\forall i \in I) (x_i \geq 0))$ then a feasible point has been obtained (this feasible point too would, in general, be nearer to the optimum point than the one obtained from the initially available infeasible point). Refer example 2 of section 6.3.

(iv) if $((\forall i \in J) (Z_j \geq 0) \& (\exists i \in I) (x_i < 0))$

then the optimal solution is obtained by the application
of either the class S decision rules of B.H.P. Method (refer (3.2.11) chapter 3) or the Dual Simplex Method (Lemke [56]).

6.2.3 Phases 1 and 2.

(i) Phase 1. Since, in considering the application of the Simplex method, the (right hand side) elements of the vector $P_0$ must be non-negative we first change the signs of all the elements of the rows corresponding to the negative (basic) variables $x_i, i \in I$.

(II) Phase 2. The criteria for the selection of the $\gamma$ and $\gamma'$ variables remain as in this phase of the Simplex Method.

It may be noted that once we enter either phase 1 or phase 2 of the Simplex Method we would not leave the latter.

The existence of the non-feasible or unbounded solution is checked according to Simplex criteria for these situations. The optimal criterion of the B.H.P. and the Simplex methods are identical.

6.2.3.1 Degeneracy. The B.H.P. method could contribute significantly to the resolution of degeneracy (or ties) in selecting the pivotal row in situations where the hyperplanes for which the ties

occur have at least one bounding hyperplane amongst them. (When this is not the case then though the degeneracy rules of B.H.P.M. could still be used by considering (3.4.3) or (3.4.6) for non-bounding hyperplanes (that is $\xi_i > 0$), their significance in resolving degeneracy is not known). Thus, for example, Beale's cycling example no. 3.7.2, chapter 3 requires, even after taking recourse to the $\varepsilon$-perturbation technique six iterations whereas by following the former technique (used in the B.H.P.M.) the optimum solution is obtained in two iterations. Moreover, as we know, the cycling phenomenon in practical problems, never occurs, but the ties or degenerate situations do occur in almost every problem. It may be worth attempting to try to see the effect on the speed of the Simplex algorithm by following the B.H.P. technique referred to above in each situation when a tie or degeneracy occurs (in different iterations).

6.3 Illustrations:

6.3.1 Example 1. Consider the example

maximise

$$ F = x_1 + 2x_2 $$

subject to

$$ x_1 + x_2 \geq 3 $$
$$ 2x_1 + x_2 \geq 4 $$
$$ -x_1 + 3x_2 \leq 12 $$
$$ 5x_1 - x_2 \leq 10 $$

and

$$ x_1, x_2 \geq 0 $$
Fig. 6.1 - Diagram corresponding to Example 1.
Table 61

Initial tableau for phase 0 of example 1.

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>j</td>
<td>i</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>C</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>C</td>
<td>1</td>
<td>2</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>C</td>
<td>1</td>
<td>2</td>
<td>-2</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>C</td>
<td>1</td>
<td>2</td>
<td>-1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>C</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ \text{Initial tableau for phase 0 of example 1.} \]
**Tableau after one iteration for phase 0/2 of example 1**

<table>
<thead>
<tr>
<th>i</th>
<th>C</th>
<th>Basis</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
<th>$P_4$</th>
<th>$P_5$</th>
<th>$P_6$</th>
<th>$P_0 (=X)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>$P_3$</td>
<td>-4/3</td>
<td>1</td>
<td>1/3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>$P_4$</td>
<td>-7/3</td>
<td>1</td>
<td>1/3</td>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>$P_2$</td>
<td>-1/3</td>
<td>1</td>
<td>1/3</td>
<td>4</td>
<td></td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>$P_5$</td>
<td>14/3</td>
<td>1</td>
<td>1/3</td>
<td>14</td>
<td></td>
<td>14</td>
<td>14</td>
</tr>
</tbody>
</table>

| $z_j$ | -5/3 | 0 | 0 | 0 | 2/3 | 0 | 8 (=F) |

**Table 62**
Rewriting the above problem as envisaged in the B.H.P.M.;
maximise
\[ F = x_1 + 2x_2 + 0x_3 + 0x_4 + 0x_5 + 0x_6 \]
subject to
\[
\begin{align*}
-x_1 - x_2 + x_3 & = -3 \\
-2x_1 - x_2 + x_4 & = -4 \\
-x_1 + 3x_2 + x_5 & = 12 \\
5x_1 - x_2 + x_6 & = 10
\end{align*}
\]
and \[ x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \]
The B.H.P.M. tableau is then as given in Table 6.1.

Phase 0:
From (3.1.21),
\[ \text{Min } (12/5, 10/3) \]
is obtained for the hyperplane \( i = \eta = 3 \)
And from (3.2.6), the axis by which to move to the bounding hyperplane \( \eta \) is given by \( \gamma = 2 \). The pivotal element is thus \( b_{32} = 3 \) as shown by the circle around it, in the above tableau.
Performing the usual Gaussian eliminational transformation, the new tableau is obtained as shown in Table 6.2.
The new point obtained is the basic feasible point \( A \) in figure 6.1.
Phase 2 of the Simplex method takes over at this point and yields the optimal point \( P (3, 5, 7, 5) \) in one step, with the optimal value of the objective function equal to 13. However, if initially phase 1 of the Simplex Method was followed (instead of
Table 6.3

Initial Tableau for phase 0 of Example 2

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>Basis</td>
<td>P_1</td>
<td>P_2</td>
<td>P_3</td>
<td>P_4</td>
<td>P_5</td>
<td>P_6</td>
<td>P_7</td>
</tr>
<tr>
<td>1</td>
<td>P_3</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>P_4</td>
<td>-2</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>P_5</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>P_6</td>
<td>5</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>P_7</td>
<td>-5</td>
<td>-2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z_j</td>
<td>-1</td>
<td>-2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

0 (=F)
Fig. 6.2 - Diagram corresponding to example 2.
Phase 0) then it would take 2 iterations to arrive at the first basic feasible point L and subsequently it takes another two iterations to reach the optimal point P. Hence, we save two iterations (one each from the operations of the phase 1 and phase 2) if we follow phase 0 prior to the applications of the Simplex Method.

6.3.2 Example 2. Consider again the example 1 with an additional constraint

\[ 5x_1 + 2x_2 \geq 9 \]

The first two B.H.P.M. tableaux are as given in Tables 6.3 and 6.4. The computations for finding the pivot are exactly the same as in Example 1. The new point obtained is still A; however, it is this time infeasible (refer figure 6.2). If we may now for the sake of simplicity of exposition, let phase 1 of the Simplex Method take over\(^{27}\), then in one step we reach the point M which is a basic feasible point; phase 2 of the Simplex Method next takes over and in one more step yields the optimal point P with optimal value equal to 13. We thus required three iterations to

\(^{26}\) The tables 6.3 and 6.4 are the same as in example 1 except for the additional row (last one) corresponding to the additional constraint in example 2 above.

\(^{27}\) If we do not let phase 1 take over at this point then the optimal point P is reached in this example in the very next step of phase 0; we, therefore, let phase 1 take over so as to illustrate the point that a 'better', infeasible point is reached by following phase 0.
Tableau after one iteration for phase 0/2 of example 2

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>P_0 (=\overline{X})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>P_3</td>
<td>-4/3</td>
<td>1</td>
<td>1/3</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>P_4</td>
<td>-7/3</td>
<td>1</td>
<td>1/3</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>P_2</td>
<td>-1/3</td>
<td>1</td>
<td>1/3</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>P_6</td>
<td>14/3</td>
<td>1/3</td>
<td>1</td>
<td>14</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>P_7</td>
<td>-17/3</td>
<td>2/3</td>
<td>1</td>
<td>-2</td>
<td>-2</td>
<td></td>
</tr>
</tbody>
</table>

C 1 2 0 0 0 0 0 0

\[ Z_j = -5/3 0 0 0 2/3 0 0 g(x-f) \]
reach the optimal point P.

On the other hand, if we had initially applied phase 1 (instead of phase 0) directly then it takes as in Example 1 two iterations to reach the basic feasible point L and an additional two iterations to reach the optimal point P using phase 2. Apart from the fact that it takes more iterations, if we use the Simplex Method alone, to reach the optimal point, the main point to be observed here is that we arrive at a 'better' basic feasible point M by following phase 0 prior to phase 1 of the Simplex Method. As indicated above it should also be noted that if we had continued to follow the B.H.P.M. alone then the optimal point P would, in this example, be reached in just two iterations.
Chapter 7

COMPARATIVE STUDY OF THE SIMPLEX AND B.H.P. METHODS

In making a comparative study we have been restricted by the number of examples available to us. Most of the examples have been drawn from the text books, different computer reference manuals and private sources. The study, though of a limited nature, however, throws significant light on the power of the B.H.P. method over the Simplex. We have attempted to compare the methods by (i) the number of iterations and (ii) the time taken to solve the different sized problems by running the two computer (all-core) programs on 16 examples.

7.1 Number of iterations required by the B.H.P.M. vis-a-vis Simplex.

Columns 3 and 4 of table 7.1 provide the detailed information in this respect. As we observe, the B.H.P. method takes less iterations in most of the examples. In none of the examples presented did we find it to take more iterations as compared to Simplex (this may however not be true generally). Furthermore the number of iterations taken by the B.H.P. method appear to reduce on comparison with Simplex method as the problem size increases. It is not known if this would generally be true. Extensive application of the method could only reveal the general situation in this respect.

7.2 Time taken by the two methods.

As could be expected the time taken by the B.H.P. method in a single iteration would, in general, be more than that required by the Simplex method. A comparison of the two 'times' given in
columns 5 and 6 of table 7.1 also points to this fact. For the (single) large problem number 16 the time taken by the B.H.P.M. is appreciably less. Again it is not known if this would generally be true. We need to run the method on very large problems to investigate this aspect fully. In general it appears, however, that the time taken by the B.H.P.M. in all cases reported in table 7.1 is either less or is comparable with the time taken by the Simplex method. It may be stated here that the total time taken by the B.H.P. method could be reduced by (i) improving the existing B.H.P.M. computer program and (ii) incorporating the techniques discussed in sections 4.2 and 4.3, chapter 4. The single iteration time though would still generally remain more than that required by the Simplex method; the B.H.P.M. however has the advantage of detecting, in general, the 'unbounded'/non-feasible' state of the given l.p. problem earlier than the Simplex method. This is because we, in each iteration, consider (i) the whole sub-space \( \mathcal{J} (j \in J_1 \subset J) \) and not just one column (of this sub-space) that is the one corresponding to \( \max |a_j| , j \in J_1 \) (as with Simplex) for unboundedness and (ii) each infeasible constraint \( i \in I_1 \subset I \) for non-feasibility (till we find at least one feasible solution point).
Table 7.1

<table>
<thead>
<tr>
<th>Serial number</th>
<th>Size of the problem ((m \times n))</th>
<th>Number of iterations taken by the two methods</th>
<th>Time taken by the two methods in secs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>(4 \times 4)</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>(4 \times 5)</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>(4 \times 5)</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>(4 \times 6)</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>(4 \times 6)</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>(5 \times 7)</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>7</td>
<td>(5 \times 7)</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>(7 \times 6)</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>(8 \times 5)</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>(8 \times 5)</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>(8 \times 5)</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>12</td>
<td>(9 \times 6)</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>(15 \times 8)</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>14</td>
<td>(19 \times 21)</td>
<td>18</td>
<td>20</td>
</tr>
<tr>
<td>15</td>
<td>(20 \times 3)</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>16</td>
<td>(54 \times 69)</td>
<td>49</td>
<td>79</td>
</tr>
</tbody>
</table>

Note 1. The size of the problem given in column 2 above does not include the slack (and surplus in case of Simplex) variables added to the constraints.

2. The times given in column 4 are obtained by running the computer program for the 'standard' Simplex method; both the times in columns 3 (inclusive of input and output times) and 4 are the execution times obtained by using a sub-routine KLOCK [37] on the I.B.M. 360/44 computer.
8.1 Extensions to quadratic programming. Consider the quadratic programming problem given by:

\[
\text{maximise } F(x) = C \times - \frac{1}{2} x^T D x, \tag{8.1.1}
\]

subject to linear inequalities

\[
A x \leq P_0, \tag{8.1.2}
\]

and

\[
x \geq 0, \tag{8.1.3}
\]

where the vectors \(C, x, P_0\) and the matrices \(A\) and \(D\) are respectively given by

\[
C = \begin{vmatrix} c_1 & c_2 & \cdots & c_n \end{vmatrix}, \tag{8.1.4}
\]

\[
X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \tag{8.1.5}
\]

\[
P_0 = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix}, \tag{8.1.6}
\]

\[
A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}. \tag{8.1.7}
\]

The quadratic programming problem will henceforth be abbreviated as 'q.p. problem'.

---

28. The quadratic programming problem will henceforth be abbreviated as 'q.p. problem'.
Further $D$ is assumed to be a symmetric positive semi-definite matrix. Introducing the $m$ non-negative slack variables $X_s$ to the inequalities (8.1.2) the above q.p. problem could be rewritten as,

maximise \[ \phi(Y) = f^T Y - \frac{1}{2} Y^T D Y, \] 8.1.9 (i)

subject to \[ BY = P_0, \] 8.1.9 (ii)

and \[ Y \geq 0, \] 8.1.9 (iii)

where the $(n+m)$ vectors $Y$ and $f$ and the $(k\times(n+m))$ matrix $B$ and $((n+m)\times(n+m))$ matrix $D$ are respectively given by

\[ Y = \begin{bmatrix} X \\ X_s \end{bmatrix}, \] 8.1.10

\[ f = \begin{bmatrix} C \\ 0 \end{bmatrix}, \] 8.1.11

\[ B = \begin{bmatrix} A & I \end{bmatrix}, \] 8.1.12
and
\[ Q = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \] (8.1.13)

Let us now consider the Kuhn-Tucker Theorem [90]. It states that the vector \( \gamma \) is a solution to the q.p. problem (8.1.9(i)-(iii)) if and only if \( \gamma \) is feasible and there exists a \((n+m)\) vector
\[ U = \begin{bmatrix} \Delta \\ \lambda \end{bmatrix} \] (8.1.14)
such that
\[ U = Q \gamma + B^T \omega - f^T \geq 0 \] (8.1.15)
and
\[ U^T \gamma = 0 \] (8.1.16)

The condition (8.1.16) implies that at most \((n+m)\) elements out of \((2n+2m)\) elements are non-zero which, in turn, means that we need to examine only basic solutions to the system (8.1.9(i)-(iii)) in our search for the optimum to the given q.p. problem (Barankin and Dorfman [4]). Thus in moving from one basic solution point to the next we enforce the condition that if a variable is in the basis then the corresponding U variable is not in the basis (the tableau corresponding to a basic solution point, with the above condition adhered to, is said to be in the 'standard form', Beale[6]).

If, therefore, a basic feasible solution is obtained such that the tableau corresponding to this basic solution is in 'standard form' and \( U \geq 0 \), then we have found the optimum point.

Let us now reconsider the optimal function (8.1.9(i)). Substituting for \( Q \gamma \) given by (8.1.15) we obtain [98]
\[
\phi(\gamma) = f^T \gamma - \frac{1}{2} \gamma^T (U - B \omega + f) \geq \frac{1}{2} f^T \gamma - \frac{1}{2} U^T \gamma + \frac{1}{2} \omega^T B \gamma,
\]
which after substitution for $BY$ from (8.1.9(ii)) gives

$$\frac{1}{2} f Y - \frac{1}{2} U^T Y + \frac{1}{2} P_o^T W,$$

that is, presuming that we consider the tableau in 'standard form'.

Thus if

$$\Phi = 2 \phi(Y),$$

we have

$$\Phi = f Y + P_o^T W,$$

Equation (8.1.15), (8.1.9(ii)) and (8.1.19) could, in view of the special structure of matrices $Q, B$ and $f$, be rewritten as

$$-DX - A^T W + C^T \leq 0,$$

$$AX \leq P_o,$$

and

$$C X + P_o^T W = F^*,$$

where

$$F^* = 2 F(X).$$

The system (8.1.20)-(8.1.22) is the usual l.p. problem and could compactly be written as

$$\begin{align*}
\text{maximise} & \quad H \xi \\
\text{subject to} & \quad G \xi \leq R \\
\text{and} & \quad \xi \geq 0
\end{align*}$$

where $H, G, R$ and $\xi$ are given by

$$H = \begin{vmatrix} C P_o^T \end{vmatrix}.$$
The dual l.p. problem to the (primal) system (8.1.24) is then given by,

\[
\begin{align*}
\text{minimise} & \quad \mathbf{R}^T \xi \\
\text{subject to} & \quad \mathbf{G}^T \xi \geq \mathbf{H}^T \\
\text{and} & \quad \xi \geq 0
\end{align*}
\]

Converting (8.1.29) to a maximising problem and the type I (\(\geq\)) inequalities in (8.1.30) to type II (\(\leq\)), the 'dual' could be rewritten as

\[
\begin{align*}
\text{maximise} & \quad -\mathbf{R}^T \xi \\
\text{subject to} & \quad -\mathbf{G}^T \xi \leq \mathbf{H}^T \\
\text{and} & \quad \xi \geq 0
\end{align*}
\]

We are now ready to apply the B.H.P.M. with restrictions that (i) the tableau is maintained in each iteration in the 'standard form' and that (ii) the hyperplane (pivotal row) selected in the immediately preceding iteration is not considered for selection in the current iteration. The latter condition may be required to be imposed in the q.p. algorithm for the B.H.P.M. on account of the former condition (so as to avoid a possible case of cycling).

Since in the primal problem the optimal solution is
obtained when we reach a basic feasible solution, the corresponding situation in the dual case is achieved when the coefficients of the functional $Z$ given by

$$Z = R$$  \hspace{1cm} (8.1.33)

are non-negative.

8.2 Accuracy in linear programming.

8.2.1 Modifications to reinversion process. Let us consider the optimal solution obtained by the application of the B.H.P.M. (or the Simplex or any other techniques) to a given l.p. problem. We know that certain constraints in the optimal point are satisfied exactly, and that the pivotal elements lie on these constraints; let these be called the pivotal constraints.

Utilising this information we reconstruct the original l.p. problem wherein the pivotal constraints are equalities and the remaining inequalities. Since we also know the pivotal elements lying on each pivotal constraint we first obtain the solution to the equality (pivotal) constraints by performing the Gaussian eliminational transformations on these pivotal elements. Next we determine the transformations to the vector $\overline{P}_v (= \overline{\beta}_{i,n+1})$ corresponding to the inequality constraints from (4.3.2) of section 4.3, chapter 4. The coefficients of the functional $Z$ will (in all probability) be non-negative to indicate that the optimal point, given by

$$\begin{pmatrix}
\overline{\beta}_{i,n+1} \\
\overline{\alpha}_{i,n+1}
\end{pmatrix}$$  \hspace{1cm} (8.2.1)

has been reached. The (new) optimal point, if different from the earlier one, would satisfy the pivotal constraints 'more'
exactly. This is on account of considerable reduction in the computations required otherwise. The (new) optimal tableau, if desired, can be obtained from (4.3.1), chapter 4. The above procedure is essentially the reinversion process [63] modified by the techniques developed in chapter 4. For further reducing the effects of rounding errors we may consider the extensions discussed generally in the section below.

8.2.2 Modifications to the equality solution techniques. Let us reconsider the techniques of section 4.2, chapter 4. We may along with finding the solution to the equality constraints obtain the inverse of the matrix corresponding to the pivotal elements and check for its accuracy by determining its product with the original matrix; the discrepancy between the latter and the unit matrix shows the extent of inaccuracy in the results.

Thus if the rank of the equality submatrix \( \mathbf{W} \) is \( n_1 \leq r \) then let \( \hat{\mathbf{W}} \) denote the leading non-singular matrix of order \( n_1 \times n_1 \) obtained, as is always possible, by rearranging the equations and unknowns (variables). Let \( \mathbf{S}_o \) denote the inverse of the submatrix \( \hat{\mathbf{W}} \) obtained as above (or by any other process of inversion) and let \( \mathbf{M}_o \) denote the discrepancy

\[
\mathbf{M}_o = \mathbf{I} - \hat{\mathbf{W}} \mathbf{S}_o, \quad \| \mathbf{M}_o \| < \rho < 1 \quad (8.2.2)
\]

29. The inverse could easily be obtained in the process of obtaining solution to the equality constraints by appending an artificial (unit) vector (as in the Simplex technique) to each of the \( m_2 \) equality constraints. The inverse corresponds to the transformations to the artificial vectors.
where \( \| M_0 \| \), considering the first matrix norm \(^{30}\) (sections 5, 13 [24]), is given by
\[
\max_k \sum_{i=1}^{n_1} | \phi_{ik} | , \quad i = 1, 2, \ldots, n_1 .
\] (8.2.3)

The accuracy of the elements of the inverse matrix \( \hat{\mathcal{W}}^{-1} \)
\( (= \theta_{ij}, \text{an } n_1 \times n_1 \text{ matrix}) \) may then be increased to as high a degree as desired by following the iterative process \([89]; \text{also sec. 13 in } [24] \) \)
\[
S_q = S_{q-1} + S_{q-1} (E - \hat{\mathcal{W}}^{-1} S_{q-1}) \]
(8.2.4)

where
\[
S_q = S_{q-1} (E + M_{q-1})
\] (8.2.5)
\[
M_q = E - \hat{\mathcal{W}}^{-1} S_q
\] (8.2.6)

(It can be proved \([24]\) that
\[
M_q = M_0^{2q}
\] (8.2.7)
(8.2.6) then reduces to
\[
S_q = \hat{\mathcal{W}}^{-1} (E - M_0^{2q})
\] (8.2.8)

which shows that \( S_q \) approaches \( \hat{\mathcal{W}}^{-1} \) within the accuracy desired.)

Having thus obtained a more accurate value for the inverse matrix \( \hat{\mathcal{W}}^{-1} \), we utilise it to obtain the final equality tableau of Section 4.2.3 by operating the former on \((a)\) those columns

30. Hotelling, however, considered the norm given by
\[
\sqrt{\sum_{i=1}^{n_1} | \phi_{ij} |^2}
\]
of the original submatrix \( W \) that have nullity \( \gamma = n_1 \) and (b) the column vector \( P_W \); thence the elements of the final equality tableau are given by

\[
\bar{\alpha}_{ij} = \sum_{k=1}^{n_1} \theta_{ik} \alpha_{kj}, \quad i \in I_W, \quad j \in J_1
\]  

(8.2.9)

and

\[
\bar{\alpha}_{i,n+1} = \sum_{k=1}^{n_1} \theta_{ik} \alpha_{kj,n+1}, \quad i \in I_W
\]  

(8.2.10)

where \( J_2 \subset J_{fa} \) is the set of indices \( j \) that correspond to (a) above, refer Section 4.4 of chapter 4. It is clear that the submatrix of the final \( (W|P_W) \) tableau corresponding to \( W \) is a unit matrix and hence these elements are not computed.

The elements of the I/F tableau are then obtained from (4.3.1 - 4.3.5) as discussed in Section 4.3 of chapter 4. The coefficients thus obtained in the final equality and I/F tables will be more accurate than the ones obtained by following the techniques of chapter 4.
BIBLIOGRAPHY


<table>
<thead>
<tr>
<th>No.</th>
<th>Reference</th>
</tr>
</thead>
</table>


<table>
<thead>
<tr>
<th>ADDENDUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>87</td>
</tr>
</tbody>
</table>