Postan 3 - Extended Postoptimal Analysis Package for Minos

G. Dobrowolski
T. Rys
K. Hayduk
A. Korytowski

December 1988
WP-88-117

Postan 3 - Extended Postoptimal Analysis Package for Minos

G. Dobrowolski
T. Rys
K. Hayduk
A. Korytowski

December 1988
WP-88-117

Foreword

This paper is one of the series of 11 Working Papers presenting the software for interactive decision support and software tools for developing decision support systems. These products constitute the outcome of the contracted study agreement between the System and Decision Sciences Program at IIASA and several Polish scientific institutions. The theoretical part of these results is presented in the IIASA Working Paper WP-88-071 entitled ***Theory, Software and Testing Examples in Decision Support Systems*** which contains the theoretical and methodological bacgrounds of the software systems developed within the project.

This paper presents the POSTAN 3 package. This package constitutes the tool for postoptimal analysis for linear and linear-fractional programming problems. POSTAN consists of a number of FORTRAN routines which are incorporated into MINOS, the well known linear and nonlinear programming code developed at the Stanford University. The postoptimal analysis is performed after MINOS has found an optimal solution and is initiated by extending the original MINOS specifiaction file. The main function of POSTAN is ranging with respect to parameters specified by the user and computing the sensitivity coefficients.

Alexander B. Kurzhanski
Chairman
System and Decision Sciences Program

POSTAN 3 – EXTENDED POSTOPTIMAL ANALYSIS PACKAGE FOR MINOS

G. Dobrowolski, T. Ryś

Joint Systems Research Department
of
Institute of Automatics,
Academy of Mining and Metallurgy, Cracow
and of
Industrial Chemistry Research Institute, Warsaw.

K. Hajduk, A. Korytowski

Institute of Automatics,
Academy of Mining and Metallurgy, Cracow.

CONTENTS

INTRODUCTION

POSTAN 3, an extended version of POSTAN [1] and POSTAN 2 [2], is a postoptimal analysis package for linear and linear-fractional programming problems. It is composed of a number of FORTRAN routines which are incorporated into MINOS, the well-known linear and nonlinear programming code developed by Murtagh and Saunders [3]. The postoptimal analysis is performed after MINOS has found an optimal solution, and is initiated by adding particular specifications to the original list of MINOS specifications.

The main objective of POSTAN 3 is ranging, i.e., determining the ranges in which certain parameters (or groups of parameters) may be changed without affecting the optimal solution and/or the optimal basis. Sensitivity coefficients which are not included in the output of the unmodified version of MINOS are also determined.

Two new auxiliary modules have been implemented in POSTAN 3 to improve its user interface:

- a module for programming a sequence of optimization problems,
- a module for decoding and selective printing of results.

A. THEORETICAL GUIDE

1. PRELIMINARY INFORMATION

The formulation of the linear problem analyzed by POSTAN is the same as for MINOS: Minimize (or maximize) a linear cost function

$$F(x) = a_0 x \tag{1}$$

subject to m row constraints:

$$d_i \le a_i x \le g_i \, , \quad i = 1,\ldots,m \tag{2}$$

and n constraints on separate variables:

$$d_{m+i} \le x_i \le g_{m+i} \, , \quad i = 1,\ldots,n \quad . \tag{3}$$

Here x is an n-dimensional column vector of decision variables, a_0 is an n-dimensional row vector of cost coefficients (also called the ***objective row***), the a_i, $i = 1,\ldots,m$, are n-dimensional row vectors, the lower bounds d_i, $i = 1,\ldots,m+n$, are real numbers or $-\infty$, and the upper bounds g_i, $i = 1,\ldots,m+n$, are real numbers or $+\infty$. Of course, if the bounds take the values $+\infty$ or $-\infty$ the corresponding relation (2) or (3) must be replaced by a strict inequality. If $d_i = g_i$, then the variable x_i is said to be ***fixed***. If $d_i = -\infty$ and $g_i = +\infty$ the variable x_i is said to be *free*. Analogous terms are used to describe the rows $a_i x$.

It should be recalled that in MINOS the two-sided inequality constraints (2) are not stated explicitly, but rather specified using ***ranges***. More precisely, a one-sided inequality is introduced in the form $a_i x \le g_i$ (type L) or $a_i x \ge d_i$ (type G), together with a real number r_i called the ***range***. In the first case, the difference between the right-hand side g_i and this number yields the lower bound ($d_i = g_i - r_i$); in the second case the sum of the right-hand side d_i and the real number r_i gives the upper bound ($g_i = d_i + r_i$).

The linear programming problem is transformed by MINOS into the following internal form: Minimize (or maximize) the variable

$$-\tilde{x}_{n+1+\mathrm{obj}} \tag{4}$$

subject to equality constraints:

$$\tilde{A}\tilde{x} = 0 \tag{5}$$

and inequality constraints:

$$\tilde{l} \le \tilde{x} \le \tilde{u} \quad . \tag{6}$$

Here $\tilde{A}$ is an $(m+1) \times (n+m+2)$-matrix:

$$\tilde{A} = \begin{bmatrix} \tilde{a}_1 & \tilde{b}_1 & \\ \cdot & \cdot & \\ \cdot & \cdot & I \\ \cdot & \cdot & \\ \tilde{a}_{m+1} & \tilde{b}_{m+1} & \end{bmatrix} \, , \tag{7}$$

where I denotes the $(m+1) \times (m+1)$ identity matrix and

$$\begin{aligned} &\tilde{a}_i = a_i \quad i < \mathrm{obj} \, , \quad \tilde{a}_{\mathrm{obj}} = a_0 \, , \quad \tilde{a}_i = a_{i-1} \quad i > \mathrm{obj} \quad , \\ &\tilde{b}_i = b_i \quad i < \mathrm{obj} \, , \quad \tilde{b}_{\mathrm{obj}} = 0 \, , \quad \tilde{b}_i = b_{i-1} \quad i > \mathrm{obj} \quad , \end{aligned} \tag{8}$$

where

$$b_i = \begin{cases} 0 \text{ if } d_i = -\infty \text{ and } g_i = +\infty \\ d_i \text{ if } d_i \text{ is finite and } g_i = +\infty \\ g_i \text{ if } g_i \text{ is finite} \end{cases} .$$

The first n components of the extended vector of decision variables $\tilde{x} \in R^{n+m+2}$ form a subvector identical to x; these components are described as ***structural***. Element $\tilde{x}_{n+1}$ is called the ***right-hand-side component***; it is fixed at -1. The remaining components of $\tilde{x}$ are called ***slack*** or ***logical components***. The objective variable $\tilde{x}_{n+1+\text{obj}}$ is free. The vector of lower bounds $\tilde{l}$ and the vector of upper bounds $\tilde{u}$ are defined as follows:

$$\tilde{l}_i = d_{m+i} \quad i = 1,\dots,n , \quad \tilde{l}_{n+1} = -1, \quad \tilde{l}_{n+1+\text{obj}} = -\infty \quad , \tag{9}$$
$$\tilde{u}_i = g_{m+i} \quad i = 1,\dots,n , \quad \tilde{u}_{n+1} = -1, \quad \tilde{u}_{n+1+\text{obj}} = +\infty \quad .$$

Now let $i = n + 1 + j$, $j = 1,\dots,m$. Then

$$\tilde{l}_i = h_i , \quad \tilde{u}_i = k_i \text{ for } j < \text{obj and } \tilde{l}_i = h_{i-1} , \quad \tilde{u}_i = k_{i-1} \text{ for } j > \text{obj} \quad , \tag{10}$$

where

$h_i = k_i = 0$ if the j-th row constraint is fixed (i.e., of type E)

$h_i = 0$, $k_i = +\infty$ if $d_j = -\infty$ and g_j is finite (one-sided constraint of type L)

$h_i = -\infty$, $k_i = 0$ if d_j is finite and $g_j = +\infty$ (one-sided constraint of type G)

$h_i = 0$, $k_i = g_j - d_j$ if d_j and g_j are finite

$h_i = -\infty$, $k_i = +\infty$ if the j-th row constraint is free .

2. POSTOPTIMAL ANALYSIS FOR LINEAR PROGRAMMING PROBLEM

This chapter presents elements of ranging theory for the linear programming problem (4)–(6). Some nonconventional notation will be used in order to avoid discussion of many particular cases. The sign $\leq$ will denote "less than or equal to" if the expressions on its both sides are finite and "less than" otherwise. Similarly, $\geq$ will denote "greater than or equal to" or "greater than". The notation $[t_1,t_2]$ will be used for the closure of the open interval (t_1,t_2); that is, t_1 and/or t_2 do not belong to the interval if they are not finite. For the sake of simplicity we shall assume that obj $= m+1$, i.e., the objective row is the last row in matrix A. As the value of variable $\tilde{x}_{n+1}$ is fixed at -1 we may remove it from the problem formulation, defining a new column vector of decision variables $y \in R^{n+m}$, where $y_i = \tilde{x}_i$ $i = n+1,\dots,n+m$. We also define an $m \in (n+m)$-matrix

$$A = \begin{bmatrix} a_1 & \\ \cdot & \\ \cdot & I \\ \cdot & \\ a_m & \end{bmatrix} ; \tag{11}$$

column vectors $b \in R^m$ (see (8)), $l,u \in R^{n+m}$, where $l_i = \tilde{l}_i$, $u_i = \tilde{u}_i$ $i = 1,\dots,n$ and $l_i = h_{i+1}$, $u_i = k_{i+1}$ $i = n+1,\dots,n+m$; and a row vector $c \in R_{n+m}$, where $c_i = a_0^i$ $i = 1,\dots,n$ and $c^i = 0$ $i = n+1,\dots,n+m$.

The linear programming problem now takes the form: Minimize (or maximize) the linear cost function

$$F(y) = cy \tag{12}$$

subject to

$$Ay = b \tag{13}$$

$$l \le y \le u \quad . \tag{14}$$

We denote the optimal solution of this problem by z and decompose it in the obvious way into the following subvectors:

z_B	basic vector
z_l	vector of nonfixed, nonbasic variables which are at their lower bounds
z_u	vector of nonfixed, nonbasic variables which are at their upper bounds
z_s	vector of fixed variables (i.e., variables for which $u_i = l_i$).

Let I_u be the set of indices of all nonbasic variables at their upper bounds and let I_l be the set of indices of all nonbasic variables at their lower bounds. Fixed variables are not included in I_u or I_l. We shall let I_B denote the set of indices of all basic variables. This decomposition is also applied to the other vectors, yielding, for example, c_B, c_l, c_u; l_B, l_l, l_u; u_B, u_l, u_u. It is clear that $z_l = l_l$, $z_u = u_u$, $z_s = u_s$. Thus the constraint matrix is decomposed into the basic matrix B and matrices L, U, S such that

$$Bz_B + Lz_l + Uz_u + Sz_s = b \quad .$$

Hence we have

$$z_B = B^{-1}b \; - \; B^{-1}(Lz_l + Uz_u + Sz_s) \tag{15}$$

for the basic vector and

$$F(z) = c_B B^{-1} b + (c_l - c_B B^{-1} L) z_1 + (c_u - c_B B^{-1} U) z_u + (c_s - c_B B^{-1} S) z_s \quad . \tag{16}$$

for the optimal cost.

Here and elsewhere we shall denote the i-th row of a matrix H by H_i and the j-th column by H^j. Define

$$D = B^{-1} \quad . \tag{17}$$

2.1. Ranging of costs

Let Δc be a given nonzero row vector in R_{n+m}, where $\Delta c^i = 0$ for $i = n+1,\ldots,n+m$ and $\Delta c^i = 0$ for fixed variables. We consider the family of linear programming problems (12)-(14) with the cost vector c replaced by $\bar{c}(t)$, where

$$\bar{c}(t) = c + t\Delta c \tag{18}$$

and t is a real number, $t \in R^1$. We wish to determine the largest range $[t_{\min}, t_{\max}]$ in which the coefficient t may vary without affecting the optimal solution, i.e., the range in which the optimal solution is equal to z for every t.

It is clear from (16) that the optimal solution remains unchanged and equal to z for all values of t such that

$$\epsilon(\bar{c}_l(t) - \bar{c}_B(t)DL) \le 0 \tag{19}$$

and

$$\epsilon(\bar{c}_u(t) - \bar{c}_B(t)DU) \ge 0 \quad , \tag{20}$$

where

$$\epsilon = \begin{cases} +1 & \text{in the case of maximization} \\ -1 & \text{in the case of minimization} \end{cases} \quad .$$

Hence

$$t\epsilon(\Delta c_l - \Delta c_B DL) \le \epsilon(c_B DL - c_l) \tag{21}$$
$$t\epsilon(\Delta c_u - \Delta c_B DU) \ge \epsilon(c_B DU - c_u) \quad .$$

We shall use the following notation:

$$T_j = -c^j + c_B DA^j \ , \quad \Delta T_j = -\Delta c^j + \Delta c_B DA^j \ , \quad j \in I_u \cup I_l \quad . \tag{22}$$

In the case of maximization we then have

$$t_{\max} = \min \{-T_j/\Delta T_j\} \quad , \tag{23}$$

where the minimum is taken over all values of j from I_l such that $\Delta T_j < 0$ and all values of j from I_u such that $\Delta T_j > 0$, and

$$t_{\min} = \max \{-T_j/\Delta T_j\} \quad , \tag{24}$$

where the maximum is taken over all values of j from I_l such that $\Delta T_j > 0$ and all values of j from I_u such that $\Delta T_j < 0$.

In the case of minimization $t_{\max}$ is determined from (23) but with the minimum taken over all values of j from I_l such that $\Delta T_j > 0$ and all values of j from I_u such that $\Delta T_j < 0$; $t_{\min}$ is determined from (24) with the maximum taken over all values of j from I_l such that $\Delta T_j < 0$ and all values of j from I_u such that $\Delta T_j > 0$.

In all cases, if the set of indices over which the maximum (or minimum) is taken is empty, then $t_{\min} = -\infty$ (or $t_{\max} = +\infty$).

From these general results it is easy to derive formulas for the cost ranging routines of POSTAN. Imposing the condition $t \ge 0$ and dropping the relations for $t_{\min}$, we obtain results suitable for the directional cost ranging routine (DIRRAN). Setting $\Delta c = e_i$, where e_i is the i-th unit vector (which has all components equal to zero except for the i-th component, which is equal to one), we obtain formulas for the ordinary cost ranging routine (CRAN). In this case we formulate the results directly in terms of the cost component $\bar{c}^i = c^i + t$. For nonbasic components we have

$$\epsilon\bar{c}^i \le \epsilon c_B DA^i \ \text{ if } \ i \in I_l \tag{25}$$

$$\epsilon\bar{c}^i \ge \epsilon c_B DA^i \ \text{ if } \ i \in I_u \tag{26}$$

If $i \in I_B$, we have, by virtue of (22):

$$\Delta T_j = D_i A^j \tag{27}$$

and

$$c^i + t_{\min} \le \bar{c}^i \le c^i + t_{\max} \quad , \tag{28}$$

where $t_{\max}$ and $t_{\min}$ are determined from (23) and (24).

At each finite boundary of the interval $[t_{\min}, t_{\max}]$ a nonbasic variable changes its state. The number of this variable and the kind of change are determined by the left-hand side component of (19) or (20) which changes its sign on the boundary. If

$$\epsilon(\bar{c}^i(t) - \bar{c}_B(t)DA^i) > 0 \quad t > t_{\max} \tag{29}$$

for some $i \in I_l$, then at the upper boundary $t = t_{\max}$ the i-th variable passes from I_l to either I_B or I_u, or the optimal solution vanishes. If (29) holds but for all $t < t_{\min}$, then an equivalent statement may be made for the lower boundary $t_{\min}$.

If

$$\epsilon(\bar{c}^i(t) - \bar{c}_B(t)DA^i) < 0 \quad t > t_{\max} \tag{30}$$

for some $i \in I_u$, then at the upper boundary $t = t_{\max}$ the i-th variable passes from I_u to either I_B or I_l, or the optimal solution vanishes. If (30) holds but for all $t < t_{\min}$, then an equivalent statement may be made for the lower boundary $t_{\min}$.

2.2. Ranging of right-hand sides

Let Δb be a given nonzero column vector in R^m. We consider the family of linear programming problems (12)–(14) with the rhs vector b replaced by $\bar{b}(t)$, where

$$\bar{b}(t) = b + t\Delta b \tag{31}$$

and $t \in R^1$. We wish to determine the largest range $[t_{\min}, t_{\max}]$ in which the coefficient t may vary without affecting the optimal basis, i.e., the range in which the optimal basis is equal to B for every t.

Letting $\bar{z}_B(t)$ denote the vector of basic variables in the optimal solution corresponding to the rhs vector $\bar{b}(t)$, we have

$$\bar{z}_B(t) = z_B + tB^{-1}\Delta b \quad . \tag{32}$$

It is clear that the nonbasic variables do not change for values of $t \in [t_{\min}, t_{\max}]$. The range $[t_{\min}, t_{\max}]$ is determined by the feasibility constraint on the basic variables:

$$l_B \le \bar{z}_B(t) \le u_B \tag{33}$$

or

$$l_B - z_B \le tD\Delta b \le u_B - z_B \quad . \tag{34}$$

Define

$$\begin{aligned}
t_1 &= \min_{j=1,\dots,m} \left\{ \frac{u_{Bj} - z_{Bj}}{D_j\Delta b} : D_j\Delta b > 0 \right\} \\
t_2 &= \max_{j=1,\dots,m} \left\{ \frac{l_{Bj} - z_{Bj}}{D_j\Delta b} : D_j\Delta b > 0 \right\} \\
t_3 &= \min_{j=1,\dots,m} \left\{ \frac{l_{Bj} - z_{Bj}}{D_j\Delta b} : D_j\Delta b < 0 \right\} \\
t_4 &= \max_{j=1,\dots,m} \left\{ \frac{u_{Bj} - z_{Bj}}{D_j\Delta b} : D_j\Delta b < 0 \right\} \quad .
\end{aligned} \tag{35}$$

We then have

$$\begin{aligned}
t_{\max} &= \min\,\{t_1, t_3\} \\
t_{\min} &= \max\,\{t_2, t_4\}
\end{aligned} \tag{36}$$

If $D_i\Delta b \le 0$ for all i, $i = 1,\dots,m$, then we set $t_1 = +\infty$ and $t_2 = -\infty$. Similarly, if $D_i\Delta b \ge 0$ for all i, $i = 1,\dots,m$, then we set $t_3 = -\infty$ and $t_4 = +\infty$.

To obtain results in a form suitable for the directional ranging routine (DRHSRN) it suffices to assume that $t \ge 0$ and to drop the relations for $t_{\min}$. To obtain formulas for the ordinary ranging routine (RHSRAN) we take $\Delta b = e_i$, where e_i is the i-th unit vector. We then have

$$D_j \Delta b = D_j^i \tag{37}$$

in (24).

At each finite boundary of the interval $[t_{\min}, t_{\max}]$ a basic variable changes its state or the optimal solution vanishes. The number j of the basic variable which becomes non-basic at the upper boundary is determined by

$$t_{\max} = \frac{u_{Bj} - z_{Bj}}{D_j \Delta b} \text{ , if } t_{\max} = t_1 \tag{38}$$

$$t_{\max} = \frac{l_{Bj} - z_{Bj}}{D_j \Delta b} \text{ , if } t_{\max} = t_3 \ . \tag{39}$$

In the first case the j-th basic variable reaches its upper bound, while in the second it passes to its lower bound. The number j of the basic variable which changes its state at the lower boundary $t = t_{\min}$ is determined by

$$t_{\min} = \frac{l_{Bj} - z_{Bj}}{D_j \Delta b} \text{ , if } t_{\min} = t_2 \tag{40}$$

$$t_{\min} = \frac{u_{Bj} - z_{Bj}}{D_j \Delta b} \text{ , if } t_{\min} = t_4 \ . \tag{41}$$

In the first case the basic variable passes to its lower bound and in the second it reaches its upper bound.

2.3. Ranging of bounds

Let col $(\Delta l, \Delta u)$ be a given column vector in $R^{2(n+m)}$,and be such that $\Delta l_i = \Delta u_i = 0$ if y_i is a fixed variable. We consider the family of linear programming problems (12) - (14) with the vectors of lower and upper bounds l and u replaced by $\bar{l}(t)$ and $\bar{u}(t)$, respectively, where

$$\bar{l}(t) = l + t\Delta l, \qquad \bar{u}(t) = u + t\Delta u \tag{42}$$

and $t \in R^1$. We wish to determine two ranges, $[t_{\text{mina}}, t_{\text{maxa}}]$ and $[t_{\text{minb}}, t_{\text{maxb}}]$. The first of these intervals is the largest range in which the coefficient t may vary without affecting the optimal solution (i.e., the range of t values for which the optimal solution remains equal to z); the second is the largest range in which t may vary without affecting the optimal basis (i.e., the range of t values for which the optimal basis remains equal to B).

The boundaries $t_{\text{mina}}, t_{\text{maxa}}$ are easily determined from the following conditions: for every $t \in [t_{\text{mina}}, t_{\text{maxa}}]$

$$\begin{aligned} & t\Delta l_i = 0 \text{ if } i \in I_l \\ & t\Delta u_i = 0 \text{ if } i \in I_u \\ & l_i + t\Delta l_i \le u_i \text{ if } i \in I_u \\ & u_i + t\Delta u_i \ge l_i \text{ if } i \in I_l \\ & l_i + t\Delta l_i \le z_i \le u_i + t\Delta u_i \text{ if } i \in I_B . \end{aligned} \tag{43}$$

The first two conditions imply that $t_{\text{mina}} = t_{\text{maxa}} = 0$ if $\Delta l_i \neq 0$ for some $i \in I_l$ and/or $\Delta u_i \neq 0$ for some $i \in I_u$.

Let $\bar{z}(t) = z + t\Delta z$ denote the optimal solution corresponding to the vector of bounds col $(\bar{l}(t), \bar{u}(t))$. Then

$$\Delta z_l = \Delta l_l \ , \ \Delta z_u = \Delta u_u \tag{44}$$

$$\Delta z_B = -D(L\Delta l_l + U\Delta u_u)$$

The values of t_{minb} and t_{maxb} may be calculated using the feasibility conditions

$$l_l + t\Delta l_l \le u_l + t\Delta u_l \ , \ l_u + t\Delta l_u \le u_u + t\Delta u_u \tag{45}$$

$$l_B + t\Delta l_B \le z_B + t\Delta z_B \le u_B + t\Delta u_B$$

or

$$t(\Delta l_l - \Delta u_l) \le u_l - l_l \tag{46}$$

$$t(\Delta l_u - \Delta u_u) \le u_u - l_u$$

$$t(\Delta l_B + DL\Delta l_l + DU\Delta u_u) \le z_B - l_B$$

$$t(\Delta u_B + DL\Delta l_l + DU\Delta u_u) \ge z_B - u_B$$

$$t_1 = \min_{j \in B} \left\{ \frac{u_j - l_j}{\Delta l_j - \Delta u_j} : \Delta l_j - \Delta u_j > 0 \right\} \tag{47}$$

$$t_2 = \max_{j \in B} \left\{ \frac{u_j - l_j}{\Delta l_j - \Delta u_j} : \Delta l_j - \Delta u_j < 0 \right\}$$

$$t_3 = \min_{j=1,\dots,m} \left\{ \frac{z_{B_j} - l_{B_j}}{\Delta l_{B_j} + D_j(L\Delta l_l + U\Delta u_u)} : \text{denominator} > 0 \right\}$$

$$t_4 = \max_{j=1,\dots,m} \left\{ \frac{z_{B_j} - l_{B_j}}{\Delta l_{B_j} + D_j(L\Delta l_l + U\Delta u_u)} : \text{denominator} < 0 \right\}$$

$$t_5 = \min_{j=1,\dots,m} \left\{ \frac{z_{B_j} - u_{B_j}}{\Delta u_{B_j} + D_j(L\Delta l_l + U\Delta u_u)} : \text{denominator} < 0 \right\}$$

$$t_6 = \max_{j=1,\dots,m} \left\{ \frac{z_{B_j} - u_{B_j}}{\Delta u_{B_j} + D_j(L\Delta l_l + U\Delta u_u)} : \text{denominator} > 0 \right\}$$

Finally,

$$t_{\text{maxb}} = \min\{t_1, t_3, t_5\} \ , \ t_{\text{minb}} = \max\{t_2, t_4, t_6\} \tag{48}$$

If the set of indices j over which a minimum or maximum is taken is empty, we substitute $+\infty$ for t_1 , t_3 , t_5 and $-\infty$ for t_2 , t_4 , or t_6 in (47). For instance, if $\Delta l_j - \Delta u_j \le 0$ for all $j \in B$, we take $t_1 = +\infty$, and so on.

Results that may be used for the directional ranging routine (DBRAN) may be obtained by assuming that $t \ge 0$ and dropping the relations for t_{mina} , t_{minb}. To obtain the formulae for the ordinary ranging routine (BRAN) we take col $(\Delta l, \Delta u) = e_i$, where e_i is the i-th unit vector in $R^{2(n+m)}$. Expressions which allow us to determine the range $[t_{\text{minb}}, t_{\text{maxb}}]$ for all types of variables are given below.

For $i \in I_l$ we define

$$t_1 = u_i - l_i \tag{49}$$

$$t_3 = \min_{j=1,\dots,m} \left\{ \frac{z_{B_j} - l_{B_j}}{D_j A^i} : D_j A^i > 0 \right\}$$

$$t_4 = \max_{j=1,\dots,m} \left\{ \frac{z_{B_j} - l_{B_j}}{D_j A^i} : D_j A^i < 0 \right\}$$

$$t_5 = \min_{j=1,\dots,m} \left\{ \frac{z_{B_j} - u_{B_j}}{D_j A^i} : D_j A^i < 0 \right\}$$

$$t_6 = \max_{j=1,\dots,m} \left\{ \frac{z_{B_j} - u_{B_j}}{D_j A^i} : D_j A^i > 0 \right\}$$

Hence

$$l_i + t_{\text{minb}} \le \bar{l}_i \le l_i + t_{\text{maxb}} \tag{50}$$

$$t_{\text{minb}} = \max\{t_4, t_6\}$$

$$t_{\text{maxb}} = \min\{t_1, t_3, t_5\}$$

and

$$\bar{u}_i \ge l_i \tag{51}$$

For $i \in I_u$ we define

$$t_1 = l_i - u_i$$

and t_3, t_4, t_5, t_6 are defined by (38). Then

$$u_i + t_{\text{minb}} \le \bar{u}_i \le u_i + t_{\text{maxb}} \tag{52}$$

$$t_{\text{minb}} = \max\{t_1, t_4, t_6\}$$

$$t_{\text{maxb}} = \min\{t_3, t_5\}$$

and

$$\bar{l}_i \le u_i \tag{53}$$

If $i \in I_B$ then

$$\bar{l}_i \le z_i \; , \; \bar{u}_i \ge z_i \tag{54}$$

At each finite boundary of the interval $[t_{\text{minb}}, t_{\text{maxb}}]$ either a basic variable changes its state or the optimal solution vanishes. If either of the first two inequalities in (45) becomes an equality at one of the boundaries, then the feasible set becomes empty at this boundary and the optimal solution vanishes. Now assume that one of the last two inequalities in (45) becomes an equality. In this case either the optimal solution vanishes or a basic variable becomes nonbasic. Let $i \in I_B$. If

$$l_i + t_{\text{minb}} \Delta l_i = z_i + t_{\text{minb}} \Delta z_i \text{ and } \Delta l_i \ne \Delta z_i$$

then at the lower boundary either the optimal solution vanishes or the i-th variable becomes nonbasic at its lower bound. Other cases may be analyzed in a similar way.

2.4. Ranging of constraints

Let Δa_i be a given row vector in R_n for some $i=1,...,m$. We consider the family of linear programming problems (12) - (14) with the i-th row of the matrix A replaced by $\bar{a}_i(t)$, where

$$\bar{a}_i(t) = a_i + t\Delta a_i \tag{55}$$

and $t \in R^1$. We wish to determine the largest open range $(t_{\min}, t_{\max})$ in which the coefficient t may vary without affecting the optimal basis or the state of nonbasic variables; more precisely, in which the sets I_B , I_l and I_u are constant. We also wish to calculate the sensitivity of the optimal cost with respect to changes of the i-th constraint

$$W = \frac{d(c\bar{z}(t))}{dt}\Big|_{t=0} \tag{56}$$

$\bar{z}(t)$ is the optimal solution of problem (11) - (14) with a_i replaced by $\bar{a}_i(t)$.

Let ue denote by I_N the set of all nonbasic variables, $I_N = \{1,...,n\} \backslash I_B$, and decompose the optimal solution $\bar{z}(t)$ into the basic vector $\bar{z}_B(t)$ and nonbasic vector $\bar{z}_N(t)$. This decomposition is also applied to other n-dimensional vectors, yielding, for example, z_B, z_N, c_B, c_N, etc. Thus, the constraint matrix A is decomposed into the basic matrix B and nonbasic matrix N. Denote by ΔA the matrix of increments of the constraint matrix A ; its i-th row is equal to Δa_i and all other rows are equal to zero. The same decomposition applied to ΔA yields the matrix of increments of the basic matrix ΔB and ΔN, the matrix of increments of N. Of course, the i-th row of ΔB is equal to Δa_{iB} , $\Delta B_i = \Delta a_{iB}$ and the i-th row of ΔN is equal to $\Delta a_{iN}, \Delta N_i = \Delta a_{iN}$. We shall also use the decomposition of A into B , U , L and S. The increments of the matrices U and L; are ΔU and ΔL only the i-th rows of ΔU and ΔL are different from zero and equal to Δa_{iu} and Δa_{il}, respectively. Moreover, we denote

$$\begin{aligned} \bar{B}(t) &= B + t\Delta B \\ \bar{N}(t) &= N + t\Delta N \\ \bar{L}(t) &= L + t\Delta L \\ \bar{U}(t) &= U + t\Delta U. \end{aligned} \tag{57}$$

We then have

$$\bar{B}(t)\bar{z}_B(t) + \bar{N}(t)\bar{z}_N(t) = b.$$

Hence we obtain for the optimal basic vector

$$\bar{z}_B(t) = (B+t\Delta B)^{-1}(Bz_B - t\Delta N z_N) \tag{58}$$

and for the optimal cost

$$F(\bar{z}(t)) = F(z) + c_B(B+t\Delta B)^{-1}(Bz_B - t\Delta N z_N) - c_B z_B \tag{59}$$

From this formula we easily derive an expression for the sensitivity

$$W = -\, c_B D \Delta A z \tag{60}$$

and for our particular form of ΔA,

$$W = -\, c_B D^i \Delta a_i z. \tag{61}$$

The range $(t_{\min}, t_{\max})$ is determined by the following conditions:

a) Nonsingularity of the basic matrix $\bar{B}(t)$

b) Bounds for the basic variables

c) Necessary conditions of optimality.

We define v^i , the representation vector of ΔB_i in the basis B,

$$v^i = \Delta B_i D. \tag{62}$$

It can be easily shown that

$$\det\bar{B}(t) = (1+tv^i)\det B \quad (B+t\Delta B)^{-1} = D - \frac{t}{1+tv^i} D^i v^i \tag{63}$$

Let us discuss condition a first. It can be written in the form

$$t \in (t_1, t_2) \tag{64}$$

where

$$t_1 = \infty\left\{ s<0 : \det\bar{B}(t)\neq 0, \text{ for every } t \in (s,0] \right\} \tag{65}$$

$$t_2 = \left\{ s>0 : \det\bar{B}(t)\neq 0, \text{ for every } t \in [0,s) \right\}.$$

From (63) it follows that

$$t_1 = -\infty, \text{ if } v^i \le 0 \text{ and } t_1 = -1/v^i, \text{ if } v^i > 0 \tag{66}$$

$$t_2 = +\infty, \text{ if } v^i \ge 0 \text{ and } t_2 = -1/v^i, \text{ if } v^i < 0.$$

Condition b has the form

$$l_B \le \bar{z}_B(t) \le u_B. \tag{67}$$

This yields in virtue of (58) and (63)

$$t(D^i \Delta a_i z - v^i z_B + v^i u_B) \ge z_B - u_B \tag{68}$$

$$t(D^i \Delta a_i z - v^i z_B + v^i l_B) \le z_B - l_B. \tag{69}$$

We denote by (t_3, t_4) the largest open interval of t values such that $0 \in (t_3, t_4)$ and conditions (68) and (69) are satisfied for every $t \in (t_3, t_4)$.

Let $\epsilon = +1$ in the problem of maximization and $\epsilon = -1$ in the problem of minimization. Condition c can be written in the form

$$\epsilon(c_l - c_B \bar{B}(t)^{-1} \bar{L}(t)) \le 0 \tag{70}$$

$$\epsilon(c_u - c_B \bar{B}(t)^{-1} \bar{U}(t)) \ge 0. \tag{71}$$

Hence, by virtue of (57) and (63) we obtain

$$\epsilon t[v^i c_l - v^i c_B DL - c_B D^i(\Delta L_i - vL)] \le \epsilon(c_B DL - c_l) \tag{72}$$

$$\epsilon t[v^i c_u - v^i c_B DU - c_B D^i(\Delta U_i - vU)] \ge \epsilon(c_B DU - c_u) \tag{73}$$

or

$$\epsilon t[v^i c^j - v^i c_B DA^j - c_B D^i(\Delta a_i^j - vA^j)] \left\{\begin{matrix} \le \\ \ge \end{matrix}\right\} \epsilon(c_B DA^j - c^j) \tag{74}$$

$j \in I_l \cup I_u$.

For every $j \in I_l$ the inequality sign is $\leq$ and for every $j \in I_u$ we take $\geq$. We denote by (t_5, t_6) the largest open interval of t values such that $0 \in (t_5, t_6)$ and conditions (72) and (73) are satisfied for every $t \in (t_5, t_6)$.

Finally, we have

$$(t_{\min}, t_{\max}) = (t_1, t_2) \cap (t_3, t_4) \cap (t_5, t_6). \tag{75}$$

These results are used in a straightforward way for the directional ranging routine ROWRAN. In order to obtain formulae for the ordinary ranging routine ELMRAN we take $\Delta a_i = e_r$, where e_r is the r-th unit vector and r is the column index of the matrix element to be analyzed. In this case we formulate the results directly in terms of the r-th element of the i-th constraint row a_i , $\bar{a}_i(t) = a_r^i + t$. The sensitivity is given by

$$W = -c_B D^i z_r. \tag{76}$$

The formulae for determining $t_{\min}$ and $t_{\max}$ are much simplified. We shall discuss them separately for the cases $r \in I_B$, $r \in I_l$, $r \in I_u$ and $r \in I_s = \{1,\ldots,n\} \backslash (I_B \cup I_l \cup I_u)$ (I_s is the set of fixed variables). Formula (75) holds in all these cases.

Assume that $r \in I_B$ and the r-th column of A is the k-th column of B. We then have $\Delta B_i = e_k$, where e_k is the k-th unit m-vector.

Hence

$$v^i = D_k^i \tag{77}$$

The condition of nonsingularity of the basic matrix $\bar{B}(t)$ takes the form (64), (66) with the substitution of (77). The bounds on basic variables yield

$$t(D^i z_r - D_k^i z_B + D_k^i u_B) \geq z_B - u_B \tag{78}$$

$$t(D^i z_r - D_k^i z_B + D_k^i l_B) \leq z_B - l_B \tag{79}$$

These conditions determine the interval (t_3, t_4). The necessary conditions of optimality (74) take the form

$$\epsilon t(D_k^i c^j - D_k^i c_B D A^j + c_B D^i D_k A^j) \begin{Bmatrix} \leq \\ \geq \end{Bmatrix} \epsilon(c_B D A^j - c^j) \tag{80}$$

for each $j \in I_l \cup I_u$.

We take $\leq$ for every $j \in I_l$ and $\geq$for every $j \in I_u$. These conditions determine the interval (t_5, t_6).

Assume now that $r \in I_l$. Then $v=0$, the basis $\bar{B}(t)$ is nonsingular for every t and so $t_1 = -\infty$, $t_2 = +\infty$. The interval (t_3, t_4) is determined by

$$z_B - u_B \leq t z_j D^i \leq z_B - l_B \tag{81}$$

and the interval (t_5, t_6) by

$$\epsilon t c_B D^i \geq \epsilon(c^r - c_B D A^r). \tag{82}$$

In the case where $r \in I_u$, we have $t_1 = -\infty$, $t_2 = +\infty$, t_3 and t_4 are determined by (81), and the interval (t_5, t_6) is calculated from the condition

$$\epsilon t c_B D^i \leq \epsilon(c^r - c_B D A^r). \tag{83}$$

If $r \in I_s$, then $t_1 = t_5 = -\infty$, $t_2, t_6 = +\infty$. The interval (t_3, t_4) is determined by (81).

At each finite boundary of the interval $(t_{\min}, t_{\max})$ either the optimal solution vanishes or a variable changes its state. The following can be said about this change at the upper boundary $t_{\max}$:

- the p-th basic variable (according to the numeration in the basis) passes to its lower bound if the p-th scalar inequality in (69) (or (79), or the right-hand side of (81)) is violated for every $t > t_{\max}$
- the p-th basic variable passes to its upper bound if the p-th scalar inequality in (69) (or (78), or the left-hand side of (81)) is violated for all $t > t_{\max}$
- the p-th (nonbasic) variable (according to the natural numeration) changes its state if inequality (74) (or (80)) for $j = p$ is violated for all $t > t_{\max}$; if (82) or (83) does not hold for all $t > t_{\max}$, then the r-th (nonbasic) variable changes its state.

The change of state at the lower boundary is determined in the same way, with $t_{\max}$ replaced by $t_{\min}$.

2.5. Ranging of columns of the constraint matrix

Let ΔA^i be a given column vector in R^m for some $i = 1, ..., n$. We consider the family of linear programming problems (12) - (14) with the i-th column of the matrix A replaced by $\bar{A}^i(t)$, where

$$\bar{A}^i(t) = A^i + t\Delta A^i \tag{84}$$

and $t \in R^1$. We wish to determine the largest open range $(t_{\min}, t_{\max})$ in which the coefficient t may vary without affecting the optimal basis or the state of nonbasic variables, that is, in which the sets I_B , I_l and I_u are constant. We also wish to calculate W, the sensitivity of the optimal cost with respect to changes of the i-th column of the constraint matrix.

We define the representation vector v ,

$$v = D\Delta A^i. \tag{85}$$

In virtue of (56) and (60) we obtain

$$W = -c_B D\Delta A^i z_i = -c_B v z_i. \tag{86}$$

Like in the previous section,

$$(t_{\min}, t_{\max}) = (t_1, t_2) \cap (t_3, t_4) \cap (t_5, t_6)$$

where (t_1, t_2) is the interval of nonsingularity of the basis $\bar{B}(t)$, (t_3, t_4) is the interval determined by the feasibility conditions on basic variables $\bar{z}(t)_B$ and (t_5, t_6) is the interval determined by the optimality conditions. For notation and precise definitions, see Section 2.4.

Let us first discuss the case where A^i is the k-th column of the basis and so $\Delta A^i = \Delta B^k$. It is easy to verify that

$$\det \bar{B}(t) = (1 + tv_k) \det B \tag{87}$$

$$\bar{B}(t)^{-1} = D - \frac{t}{1 + tv_k} v D_k$$

We therefore have

$$t_1 = -\infty, \text{ if } v_k \leq 0 \text{ and } t_1 = -1/v_k, \text{ if } v_k > 0 \tag{88}$$

$$t_2 = +\infty, \text{ if } v_k \geq 0 \text{ and } t_2 = -1/v_k, \text{ if } v_k < 0.$$

The condition of feasibility of the basic vector (67) yields

$$t(vz_i - v_k z_B + v_k u_B) \geq z_B - u_B \tag{89}$$

$$t(vz_i - v_k z_B + v_k l_B) \leq z_B - l_B. \tag{90}$$

(t_3,t_4) is the largest open interval of t values such that $0 \in (t_3,t_4)$ and inequalities (89) and (90) are satisfied for every $t \in (t_3,t_4)$. The necessary conditions of optimality (70) and (71) take the form

$$\epsilon t(v_k c^j - v_k c_B D A^j + c_B v D_k A^j) \begin{Bmatrix} \leq \\ \geq \end{Bmatrix} \epsilon(c_B D A^j - c^j) \tag{91}$$

for *each* $j \in I_l \cup I_u$

where we take the sign $\leq$ for every $j \in I_l$ and $\geq$ for every $j \in I_u$. (t_5,t_6) is the largest open interval of t values such that $0 \in (t_5,t_6)$ and inequalities (91) hold for every $t \in (t_5,t_6)$.

Assume now that A^i is a nonbasic column. Then $t_1 = -\infty$ and $t_2 = +\infty$. The feasibility conditions for $\bar{z}_B(t)$, and so the formulae for determining t_3 and t_4 do not depend on whether i belongs to I_l , I_u or I_s. The feasibility conditions (67) yield

$$z_B - u_B \leq t v z_i \leq z_B - l_B. \tag{92}$$

The interval (t_3,t_4) is defined as before. If $i \in I_s$, then $t_5 = -\infty$ and $t_6 = +\infty$. If $i \in I_l$, then the optimality conditions (70) and (71) give

$$\epsilon t c_B v \geq \epsilon(c^i - c_B D A^i). \tag{93}$$

If $i \in I_u$, we obtain from (70) and (71)

$$\epsilon t c_B v \leq \epsilon(c^i - c_B D A^i). \tag{94}$$

Inequalities (93) and (94) allow us to determine the interval (t_5,t_6) which is defined as before.

At each finite boundary of the interval $(t_{\min},t_{\max})$ either the optimal solution vanishes or a variable changes its state.

The following can be said about this change at the upper boundary $t_{\max}$:

- the p-th basic variable (according to the numeration in the basis) passes to its lower bound if the p-th scalar inequality in (90) (or the right-hand side of (92)) is violated for all $t > t_{\max}$
- the p-th basic variable (according to the numeration in the basis) passes to its upper bound if the p-th scalar inequality in (90) (or the left-hand side of (92)) is violated for all $t > t_{\max}$
- the p-th (nonbasic) variable (according to the natural numeration) changes its state if inequality (91) for $j=p$ is violated for all $t > t_{\max}$; if (93) or (94) does not hold for all $t > t_{\max}$, then the i-th (nonbasic) variable changes its state.

The change at the lower boundary is determined in a similar fashion, $t_{\max}$ replaced by $t_{\min}$.

3. POSTOPTIMAL ANALYSIS FOR LINEAR - FRACTIONAL PROGRAMMING PROBLEM

3.1. Solution of linear-fractional problem using the simplex algorithm

Linear-fractional programming LFP problem may be defined as follows:

$$\inf\left\{ \frac{cx+\alpha}{dx+\beta} \;:\; Ax \le b,\; x \ge 0 \right\} \tag{95}$$

where

$x \in R^n$, $b \in R^m$ are column vectors; c, $d \in R_n$ are row vectors; $\alpha, \beta \in R$ and A is an mxn matrix.

Solving algorithms of the fractional programming that are dealt with in literature may be divided into 4 groups depending on the general strategy of problem solving. However all of those use, in effect, well-known and checked numerical algorithms.

The possibility chosen for POSTAN 3 is direct solving the linear-fractional problem using the modified simplex algorithm. Several reasons can be given to justify this decision:

1. As will be shown instantly, the LFP problem has similar properties as LP, thus can be represented in a similar canonical form in the solving algorithm.
2. Following above, the cost of implementation is relatively low.
3. The similarities help to implement postoptimal analysis for the LFP case.
4. As a by-product, controlling the algorithm will be similar for both cases and it will be possible to switch the mode of operation (LP - LFP) for the problem already entered.

We assume that the vector x has an additional component equal to 1 which corresponds to free terms of numerator and denominator. Having the numerator $N(x) = c_0 x$ and the denominator $M(x) = d_0 x$ we rewrite LFP problem in the internal MINOS form:

Minimize (or maximize) the quotient of variables:

$$-\tilde{x}_{n+1+\text{nom}} \quad , \quad -\tilde{x}_{n+1+\text{den}} \tag{96}$$

subject to equality constraints:

$$\tilde{A}\tilde{x} = 0 \tag{97}$$

and inequality constraints:

$$\tilde{l} \le \tilde{x} \le \tilde{u} \quad . \tag{98}$$

Here $\tilde{A}$ is an $(m+2) \times (n+m+3)$-matrix:

$$\tilde{A} = \begin{bmatrix} \tilde{a}_1 & \tilde{b}_1 & \\ \cdot & \cdot & \\ \cdot & \cdot & I \\ \cdot & \cdot & \\ \tilde{a}_{m+1} & \tilde{b}_{m+1} & \\ \tilde{a}_{m+2} & \tilde{b}_{m+2} & \end{bmatrix} , \tag{99}$$

where I denotes the $(m+2) \times (m+2)$ identity matrix and

$$\tilde{a}_i = a_i \quad i < \text{obj} \ , \quad \tilde{a}_{\text{nom}} = c_0 \ , \quad \tilde{a}_{\text{den}} = d_0 \ , \quad \tilde{a}_i = a_{i-2} \quad i > \text{den} \quad , \qquad (100)$$
$$\tilde{b}_i = b_i \quad i < \text{obj} \ , \quad \tilde{b}_{\text{nom}} = 0 \ , \quad \tilde{b}_{\text{den}} = 0 \ , \quad \tilde{b}_i = b_{i-2} \quad i > \text{den} \quad ,$$

where

$$b_i = \begin{cases} 0 \text{ if } d_i = -\infty \text{ and } g_i = +\infty \\ d_i \text{ if } d_i \text{ is finite and } g_i = +\infty \\ g_i \text{ if } g_i \text{ is finite} \end{cases} .$$

For the sake of simplicity, it is assumed above that the rows of the numerator and denominator are placed in matrix $\tilde{A}$ on last two positions indexed by $nom = m + 1$ and $den = m + 2$, respectively. Neglecting the order of rows, the representation of constraints is analogous as in LP, see equations (9) and (10).

Let us now assume that $M(x) > 0$ for each admissible x and thus a few of properties of the problem LFP can be written out *per analogiam* to the LP case. They are important for the construction of the solving algorithm.

1. The admissible set is a closed and convex polyhedron.
2. As the denominator is assumed to be positive the criterion is continuous and reaches its minimum and maximum if the admissible set is bounded.
3. The criterion (quotient) is strictly quasi convex (strictly quasi concave) and each of its local minimum (maximum) is also global.
4. The criterion reaches its minimum (maximum) at a vertex of the admissible set.
5. The criterion has planes of constant value.
6. If the minimum (maximum) is reached in more than one vertex it is reached at any boundary point of the admissible set being linear combination of these vertex points.

Further on the simplex algorithm course will be shown for the LFP case.

In phase I of the algorithm an admissible solution is searched for. Since both in LP and LFP problems the definition of admissible solutions are identical, the whole phase I may be used without any changes.

In phase II the simplex algorithm iteratively passes from a current basic solution to the consecutive one with smaller (greater) value of the objective function.

This is done in the following steps:

1. A variable is chosen for which a relative improvement of the objective function is attained and it is entered into the basis.
2. A variable is found which leaves the basis, i.e., reaches its upper or lower limit as the first from among all basic variables which change their values according to change of variable entering the basis.
3. A new basic matrix is computed after the exchange. This operation is not performed in the case when the variable chosen jumps from one of its limits to the opposite not affecting the status of the others.

The above procedure is interrupted when one of the stopping conditions is fulfilled:

1. The admissible set is empty - this is detected in phase I.
2. The optimal solution is reached - introduction of any nonbasic variables does not improve the objective function.

3. The solution is unbounded - the variable introduced into the basis may unlimitedly change not affecting the status of the others.
4. The solution is unbounded - the denominator can attain a zero value (a condition specific for the LFP implementation). Preserving $M(x){>}0$ is traced.

It results from the properties of the problem that only step 1 of phase II of the simplex algorithm has to be modified as two others do not directly depend on the form of objective function.

In the simplex algorithm the criterion of choice of the variable that enters the basis is the reduced cost calculated for nonbasic variables. It is calculated using the current basic matrix and original coefficients of the problem and is equal to the partial derivative of objective function with respect to the given variable at the current vertex point. Due to linearity of the objective function it ensures decreasing (increasing) the objective function in consecutive vertex pointed at by the unit vector corresponding to the given variable.

Thus the following lemma is essential for the described modification of the simplex algorithm.

Lemma. Let

$$h(\alpha) = \frac{c(x^0 + \alpha\xi)}{d(x^0 + \alpha\xi)} \tag{101}$$

where: x^0, $\xi \in R^n$, $c,d \in R_n$ and α is a real. Assume that the set

$$H = \{\alpha \in R^1 : d(x^0 + \alpha\xi) > 0\} \tag{102}$$

is nonempty.

The function h is continuous and differentiable on H and its derivative h' is of constant sign on H. In particular, if $h'(\alpha_1)<0$ for some $\alpha_1 \in H$, then $h'(\alpha)<0$ for every $\alpha \in H$.

The modified simplex algorithm computes modified reduced cost for nonbasic variables with respect to the linear-fractional objective function i.e. its corresponding directional derivatives. The modified reduced cost for the j-th nonbasic variable is as follows:

$$\frac{M(x^0)\,(c_j - c_B B^{-1} A^j) - N(x^0)\,(d_j - d_B B^{-1} A^j)}{M^2(x^0)} \tag{103}$$

Having in mind that $-c_B B^{-1}$ and $-d_B B^{-1}$ are the simplex multipliers of current basis with respect to $N(x)$ and $M(x)$, the modified reduced cost in LFP case (103) can be computed from the simplex multipliers, coefficients of the original matrix of the problem and the values of numerator and denominator.

It can be easily checked that the modified reduced cost computed according to (103) is appropriate as the criterion of choice of a variable entering the basis. Therefore, the optimality test may be performed as in the original simplex algorithm. Similarly, other stopping conditions of the algorithm are independent from that modification.

The notation of Chapter 2 will be introduced here for the LFP case.

Minimize (or maximize) the quotient of linear functions:

$$N(y) = cy \quad , \quad M(y) = dy \tag{104}$$

subject to

$$Ay = b \tag{105}$$

$$l \leq y \leq u \quad . \tag{106}$$

We denote the optimal solution of this problem by z and decompose it in the obvious way into the following subvectors:

z_B	basic vector
z_l	vector of nonfixed, nonbasic variables which are at their lower bounds
z_u	vector of nonfixed, nonbasic variables which are at their upper bounds
z_s	vector of fixed variables (i.e., variables for which $u_i = l_i$).

Let I_u be the set of indices of all nonbasic variables at their upper bounds and let I_l be the set of indices of all nonbasic variables at their lower bounds. Fixed variables are not included in I_u or I_l. We shall let I_B denote the set of indices of all basic variables. This decomposition is also applied to the other vectors, yielding, for example, c_B, c_l, c_u; l_B, l_l, l_u; u_B, u_l, u_u. It is clear that $z_l = l_l$, $z_u = u_u$, $z_s = u_s$. Thus the constraint matrix is decomposed into the basic matrix B and matrices L, U, S such that

$$Bz_B + Lz_l + Uz_u + Sz_s = b \quad .$$

Hence we have

$$z_B = B^{-1}b - B^{-1}(Lz_l + Uz_u + Sz_s) \tag{107}$$

for the basic vector and

$$\frac{N(z)}{M(z)} = \frac{c_B B^{-1}b + (c_l - c_B B^{-1}L)z_l + (c_u - c_B B^{-1}U)z_u + (c_s - c_B B^{-1}S)z_s}{d_B B^{-1}b + (d_l - d_B B^{-1}L)z_l + (d_u - d_B B^{-1}U)z_u + (d_s - d_B B^{-1}S)z_s} \tag{108}$$

for the optimal cost.

3.2. Ranging of cost in LFP case

Let Δc, Δd be given row vectors in R_{n+m} (not equal to zero simultaneously) where Δc^i, $\Delta d^i = 0$ for $i = n+1,\ldots,n+m$ and for fixed variables. We consider the family of LFP problems (104)-(106) with the cost vectors c, d replaced by $\bar{c}(t)$ and $\bar{d}(t)$, respectively, where:

$$\bar{c}(t) = c + t\Delta c \quad , \quad \bar{d}(t) = d + t\Delta d \tag{109}$$

and t is a real number, $t \in R^1$. We wish to determine the largest range $[t_{\min}, t_{\max}]$ in which the coefficient t may vary without affecting the optimal solution, i.e., the range in which the optimal solution is equal to z for every t.

It is clear from (108) that the optimal solution remains unchanged and equal to z for all values of t such that

$$\epsilon(M(z,t)\ (\bar{c}_l(t) - \bar{c}_B(t)DL) - N(z,t)\ (\bar{d}_l(t) - \bar{d}_B(t)DL)) \leq 0 \tag{110}$$

and

$$\epsilon(M(z,t)\ (\bar{c}_u(t) - \bar{c}_B(t)DU) - N(z,t)\ (\bar{d}_u(t) - \bar{d}_B(t)DU)) \geq 0 \tag{111}$$

where

$$\epsilon = \begin{cases} +1 & \text{in the case of maximization} \\ -1 & \text{in the case of minimization} \end{cases}$$

and as long as the denominator will be positive

$$M(z,t) > 0 \quad . \tag{112}$$

These general results are analogous to those obtained for LP case. From among a few possibilities of postoptimal analysis that arise here:

1. Varying just one cost coefficient no matter of the numerator nor denominator as in the ordinary ranging routine for LP case.
2. Varying a pair of numerator and denominator coefficients corresponding to the same variable.
3. Varying all coefficients in the fashion of the directional cost routine for LP case.

the first one is implemented for POSTAN 3.

Setting $\Delta c = e_i$ and $\Delta d = 0$ (or in the opposite manner), where e_i is the i-th unit vector (which has all components equal to zero except for the i-th component, which is equal to one), we obtain formulas for the ordinary cost ranging routine (CRAN) in the LFP case. In this case we formulate the results separately for the numerator and denominator cost components.

We shall use the following notation:

$$T_i^c = c^i - c_B DA^i \ , \qquad (113)$$
$$T_i^d = d^i - d_B DA^i \ .$$

For nonbasic components $(i \in I_u \cup I_l)$ of the numerator we have

$$t_i^c = \frac{N(z,0)\ T_i^d - M(z,0)\ T_i^c}{-z_i\ T_i^d + M(z,0)} \qquad (114)$$

and for the denominator components

$$t_i^d = \frac{-\ N(z,0)\ T_i^d + M(z,0)\ T_i^c}{N(z,0)\ -\ z_i\ T_i^c} \qquad (115)$$

together with

$$t_i^M = \frac{-\ M(z,0)}{z_i} \qquad (116)$$

In terms of intervals for a numerator cost component corresponding to the nonbasic variable we obtain:

$$\bar{c}^i \in \begin{cases} [c^i + t_i^c, +\infty) & \text{for} \quad t_i^c < 0 \\ (-\infty\,, c^i + t_i^c] & \text{for} \quad t_i^c > 0 \end{cases} \qquad (117)$$

and for a denominator cost component:

$$\bar{d}^i \in A \cap B \qquad (118)$$

where

$$A = \begin{cases} [d^i + t_i^d, +\infty) & \text{for} \quad t_i^d < 0 \\ (-\infty\,, d^i + t_i^d] & \text{for} \quad t_i^d > 0 \end{cases}$$

$$B = \begin{cases} (t_i^M, +\infty) & \text{for} \quad t_i^M < 0 \\ (-\infty\,, t_i^M) & \text{for} \quad t_i^M > 0 \end{cases}$$

For primarily degenerated variables formulas (110) and (111) become equalities. Then the boundary value of parameter t is equal to zero and the range for the parameter depends on the status (nonbasic at lower or upper limit) of the particular variable given by the simplex algorithm.

For the numerator basic components ($i \in I_B$) the formulas are a bit complicated:

$$t_{ij}^c = \frac{-N(z,0)\ T_j^d + M(z,0)\ T_j^c}{z_i\ T_j^d + M(z,0)\ D_i A^j}\ ,\quad j \in I_u \cup I_l \tag{119}$$

and

$$t_{\min c}^i = \min\{t_{ij}^c : t_{ij}^c < 0\}$$
$$t_{\max c}^i = \max\{t_{ij}^c : t_{ij}^c > 0\}$$

and for the denominator basic components in presence of the condition (116)

$$t_{ij}^d = \frac{N(z,0)\ T_j^d - M(z,0)\ T_j^c}{N(z,0)\ D_i A^j + z_i\ T_j^c}\ ,\quad j \in I_u \cup I_l \tag{120}$$

and

$$t_{\min d}^i = \min\{t_{ij}^d : t_{ij}^d < 0\}$$
$$t_{\max d}^i = \max\{t_{ij}^d : t_{ij}^d > 0\}$$

In terms of intervals the results look as follows:

$$\bar{c}^i \in [c^i - t_{\min c}^i\, , c^i + t_{\max c}^i] \tag{121}$$
$$\bar{d}^i \in [d^i - t_{\min d}^i\, , d^i + t_{\max d}^i] \cap B \tag{122}$$

where set B is defined as in (118).

In all these cases, if the set of indices over which the maximum (or minimum) is taken is empty, then $t_{\min} = -\infty$ (or $t_{\max} = +\infty$).

At each finite boundary of the interval $[t_{\min}, t_{\max}]$ a nonbasic variable changes its state. The number of this variable and the kind of change are determined by the left-hand side component of (110) or (111) or (112) which changes its sign on the boundary. If

$$\epsilon(M(z,t)\ (\bar{c}_l^i(t) - \bar{c}_B(t)DA^i) - N(z,t)\ (\bar{d}_l^i(t) - \bar{d}_B(t)DA^i)) > 0 \quad t > t_{\max} \tag{123}$$

for some $i \in I_l$, then at the upper boundary $t = t_{\max}$ the i-th variable passes from I_l to either I_B or I_u, or the optimal solution vanishes. If (120) holds but for all $t < t_{\min}$, then an equivalent statement may be made for the lower boundary $t_{\min}$. If

$$\epsilon(M(z,t)\ (\bar{c}_u^i(t) - \bar{c}_B(t)DA^i) - N(z,t)\ (\bar{d}_u^i(t) - \bar{d}_B(t)DA^i)) < 0 \quad t > t_{\max} \tag{124}$$

for some $i \in I_u$, then at the upper boundary $t = t_{\max}$ the i-th variable passes from I_u to either I_B or I_l, or the optimal solution vanishes. If (124) holds but for all $t < t_{\min}$, then an equivalent statement may be made for the lower boundary $t_{\min}$.

B. USER MANUAL

1. IMPLEMENTATION OF LINEAR POSTAN IN MINOS

In order to insert the POSTAN procedures into MINOS, and to allow them to be used in the same way as other MINOS facilities, we have made the changes outlined below.

1.1. New key-words in the SPECS file

Key	default	meaning
BYELEMS **COST** RANGING	off	This activates the postoptimal analysis of cost ranges. Subroutine CRAN is called (see Section 2.1). Insensitivity ranges for each cost coefficient are calculated under the assumption that the values of the others are kept constant. There is no request for data
BYELEMS **RHS** RANGING	off	A similar procedure is carried out for each component of the rhs vector. Subroutine RHSRAN is called (see Section 2.2). There is no request for data
BYELEMS **BOUND** RANGING	off	This command initiates the computation of insensitivity ranges for the upper and lower bounds of each structural and logical variable. Subroutine BRAN is called (see Section 2.3). Insensitivity ranges are produced for two cases: NO SOLUTION CHANGE and NO BASIS CHANGE. There is no request for data
MATRIX **ELE**MENT RANGING	off	This activates postoptimal analysis of elements of the constraint matrix. Subroutine ELMRAN is called (see Section 2.4). The list of elements has to be specified
DIRECTIONAL **COST** RANGING	off	This is the first of five commands which enable the user to perform directional postoptimal analysis. The cost vector is shifted along the direction indicated by the data. Subroutine DIRRAN is called (see Section 2.5). The length of insensitivity interval thus obtained is printed out
DIRECTIONAL **RHS** RANGING	off	This command activates postoptimal analysis of the rhs vector. Subroutine DRHSRN is called (see Section 2.6). The direction of change has to be specified
DIRECTIONAL **BOUND** RANGING	off	Postoptimal analysis of the upper and lower bounds of all variables is activated. Subroutine DBRAN is called (see Section 2.7). The user has to provide data to define the direction of alteration of both upper and lower bounds

New key-words in SPECS file (cont.)

Key	default	meaning
MATRIX ROW RANGING	off	This activates postoptimal analysis of a constraint row. The row number and the direction of alteration have to be specified by the user. Subroutine ROWRAN is called (see Section 2.8)
MATRIX COLUMN RANGING	off	Postoptimal analysis of a column of the constraint matrix is initiated. The user has to specify the column number and the direction of alteration. Subroutine COLRAN is called (see Section 2.9)
DATA RANGING FILE n	$n = 5$	This key-word specifies the logical number of the data file for procedures of matrix elements ranging and directional ranging. If none of these procedures is called, this key-word will be ignored if it is present. It is obligatory to declare a data ranging file if a procedure which requires data is used
PRINT DATA RANGING FILE	none	If this key-word is used, the whole Data Ranging File will be printed in the output. Otherwise, only the records with comments and the records NAME, SET and ENDATA are printed

The parts of the key-words which are analyzed by the program, are printed in **bold** font.

1.2. Data ranging file – input format

The data for the postoptimal analysis procedures are prepared in an MPS-like format and placed in the file specified by the MINOS key-word DATA RANGING FILE. The data sets for different directional ranging procedures may be given in any order. The beginning of the data set for each procedure is identified by the line NAME and its end by the line ENDATA. The line 'SET' may occur immediately after the line NAME in each data set; this line defines the default values of all the variables which are not explicitly defined. Every data set is identified by the name given in the line NAME.

The records in the DATA RANGING FILE should have the following (basic) form, which is analogous to MPS format:

Columns:	1-4,	5–12,	15–22,	25–36,	40–47,	50–61
Fields:	f1,	f2,	f3,	f4,	f5,	f6

Below we give a detailed description of the data set for each directional ranging procedure and for ELMRAN.

Directional Bound Ranging

	f1	f2	f3	f4	f5	f6
1.	NAME		DBOU			
2.	'SET'	Comments		Value		
3.		LOWER	Row/col. name	Value	Row/col. name	Value
4.		UPPER	Row/col. name	Value	Row/col. name	Value
5.	ENDATA					

Remarks:

- If field f2 in a given record is empty, this means that it is the same as in the last record. Field f2 must not be empty in the first data record.
- The records with identifiers UPPER and LOWER may appear in an arbitrary order.
- LOWER is used for increments of the lower bounds and UPPER for increments of the upper bounds.

Directional Cost Ranging

	f1	f2	f3	f4	f5	f6
1.	NAME		DCOS			
2.	'SET'	Comments		Value		
3.			Col. name	Value	Col. name	Value
4.	ENDATA					

Directional RHS Ranging

	f1	f2	f3	f4	f5	f6
1.	NAME		DRHS			
2.	'SET'	Comments		Value		
3.			Row name	Value	Row name	Value
4.	ENDATA					

Matrix Element Ranging

	f1	f2	f3	f4	f5	f6
1.	NAME		MELE			
2.		Col. name	Row name		Row name	
3.	ENDATA					

Remarks:

- Field f2 of record 2 may be empty. This means that it is the same as in the previous record. It must not be empty in the first data record.
- The maximum number of elements which can be specified is the integer part of $N/2$ where N is the number of all variables, $N = n + m + 2$.

Matrix Row Ranging

	f1	f2	f3	f4	f5	f6
1.	NAME		MROW			
2.	'SET'	Comments		Value		
3.		Row name	Col. name	Value	Col. name	Value
4.	ENDATA					

Matrix Column Ranging

	f1	f2	f3	f4	f5	f6
1.	NAME		MCOL			
2.	'SET'	Comments		Value		
3.		Col. name	Row name	Value	Row name	Value
4.	ENDATA					

Remarks:

- In both data sets, field f2 of record 3 may be empty; this means that it is the same as in the previous record. Field f2 must not be empty in the first data record.
- Only one row and/or column may be specified.

The following general rules apply to all data sets:

- One of the pairs of fields (f3,f4) and (f5,f6) may be empty.
- If 'SET' appears, it must follow immediately after NAME. If 'SET' does not occur, the default for all variables whose values are not specified is zero. This has the same effect as

  'SET'   0.
- Comments may be entered in arbitrary positions in the data set. They are identified by an asterisk * in the first column.
- The values should be written as real numbers in a format accepted by FORTRAN
- DATA RANGING FILE is read once to find the necessary data set (one cycle is performed). This file is not rewound if it is correctly constructed and the data sets occur in the following order: DCOS, DBOU, DRHS, MELE, MROW, MCOL.

1.3. Subroutines of linear POSTAN

In its present form the package contains nine subroutines, which can be divided into two groups. CRAN, RHSRAN, BRAN and ELMRAN perform ordinary ranging (by elements) while DIRRAN, DRHSRN, DBRAN, ROWRAN and COLRAN perform directional ranging. In this section we describe the output produced by each subroutine and give an explanation of the results. The inputs are described in Section 1.2 and the mathematical theory is presented in Section A.2 .

1.3.1. CRAN

CRAN performs ordinary ranging on the costs. For each cost component a_0^i, $i = 1,...,n$, the subroutine determines the largest range in which a_0^i may vary without affecting the optimal solution. While the range for a_0^i is being determined, all other components a_0^j, $j \neq i$, remain fixed at their original values. CRAN also gives some information on the change of state of variables at the boundaries.

This subroutine does not require any input data.

The output is entitled COST RANGING. The following information is then given for each cost component, $i = 1,...,n$:

NUMBER	Number of structural variable
COLUMN	Name of structural variable
OBJ GRADIENT	Cost component
LOWER LIMIT	Lower boundary of the range in which the cost component may vary without affecting the optimal solution
UPPER LIMIT	Upper boundary of this range

CHANGE AT LOWER LIMIT (OR OPTSOL VANISHES)	Name of the nonbasic variable which changes its state at the lower boundary; this is printed only if the lower boundary is finite. (Beware: CRAN does not know if there is an optimal solution beyond the boundary so that the name of a nonbasic variable may be printed even if the optimal solution vanishes)
CHANGE AT UPPER LIMIT (OR OPTSOL VANISHES)	Name of the nonbasic variable which changes its state at the upper boundary (other explanations as above)
M+J	NUMBER $+ m + 1$

1.3.2. RHSRAN

RHSRAN performs ordinary ranging on the right-hand sides (rhs). For each component $\tilde{b}_i$, $i = 1,...,m+1$, of the vector of right-hand sides (except for the objective row, $i \neq$ obj), this subroutine determines the maximum range in which $\tilde{b}_i$ may vary without affecting the optimal basis. While the range for $\tilde{b}_i$ is being determined, all other components $\tilde{b}_j$, $j \neq i$, are fixed at their original values. It should be noted that the rhs vector $\tilde{b}$ is not always the right-hand side of a constraint system in the original formulation (1)–(3); the user should refer to (5)–(10). In addition, RHSRAN gives some information on the change of state of variables at the boundaries.

This subroutine does not require any input data.

The output is entitled RHS RANGING. The following information is then given for each rhs component, $i = 1,...,m+1$, except for the objective row, $i \neq$ obj:

NUMBER	$n + i + 1$
ROW	Name of row
RHS	Right-hand-side component $\tilde{b}_i$
LOWER LIMIT	Lower boundary of the range in which the rhs component may vary without affecting the optimal basis
UPPER LIMIT	Upper boundary of this range
CHANGE AT LOWER LIMIT (OR OPT SOL VANISHES)	Name of the basic variable which becomes nonbasic at the lower boundary. LL is printed if this variable reaches its lower bound and UL if it reaches its upper bound; the name is printed only if the boundary is finite. (Beware: RHSRAN does not know if there is an optimal solution beyond the boundary and so a variable name may be printed even if the optimal solution vanishes)
CHANGE AT UPPER LIMIT (OR OPT SOL VANISHES)	Name of the basic variable which becomes nonbasic at the upper boundary (other explanations as above)
M+J	Number of row

1.3.3. BRAN

BRAN performs ordinary ranging on the bounds. For each lower bound $\tilde{l}_i$ and each upper bound $\tilde{u}_i$, $i = 1,...,n+m+2$, the subroutine determines two ranges: range A, which is the maximum range in which the bound may vary without affecting the optimal solution, and range B, which is the maximum range in which the bound may vary without affecting the optimal basis. While these ranges are being determined for $\tilde{l}_i$ (or $\tilde{u}_i$), all other bounds remain fixed at their original values. BRAN also gives some information on the change of state of variables at the boundaries. This analysis is not performed for fixed variables, i.e., if $\tilde{u}_i = \tilde{l}_i$.

This subroutine does not require any input data.

The output is entitled BOUND RANGING. It is divided into two parts, A and B, which will now be discussed separately.

Part A

Part A is entitled A. NO SOLUTION CHANGE and is divided into two subsections, SECTION 1 – ROWS and SECTION 2 – COLUMNS, which correspond to the sections of the same name in the final output of MINOS.

SECTION 1 – ROWS contains the following information for each slack variable $\tilde{x}_i$, $i = n+2,...,n+m+2$ (or for each row constraint), except for the slack variable $\tilde{x}_{n+1+\mathrm{obj}}$ which corresponds to the objective row. In the first two columns we have:

NUMBER	Number of slack variable i
ROW	Name of row

If $\tilde{u}_i = \tilde{l}_i$ for the slack variable under consideration, the remaining columns contain only the message FIXED VARIABLE.

In the case when the slack variable $\tilde{x}_i$ is nonbasic at its lower bound the message VARIABLE AT LOWER BOUND appears in the next two columns, which otherwise contain:

LL FOR L BOUND	Lower boundary of range A for $\tilde{l}_i$
UL FOR L BOUND	Upper boundary of range A for $\tilde{l}_i$

The next two columns give similar information about the upper bound. In other words, if the slack variable $\tilde{x}_i$ is nonbasic at its upper bound, the message VARIABLE AT UPPER BOUND is printed; if not the columns contain:

LL FOR U BOUND	Lower boundary of range A for $\tilde{u}_i$
UL FOR U BOUND	Upper boundary of range A for $\tilde{u}_i$

The last column contains:

I	Row number

SECTION 2 – COLUMNS contains information analogous to that described above for each structural variable $\tilde{x}_i$, $i = 1,...,n$. All of the information may be interpreted in the same way as in SECTION 1 – ROWS, with the following exceptions:

NUMBER	Number of structural variable
COLUMN	Name of structural variable
M+J	$m+1+i$

Part B

Part B is entitled B. NO BASIS CHANGE. It is also divided into two subsections, SECTION 1 - ROWS and SECTION 2 - COLUMNS.

SECTION 1 - ROWS contains the following information for each slack variable $\tilde{x}_i$, $i = n+2,...,n+m+2$, except for the slack variable $\tilde{x}_{n+1+\mathrm{obj}}$ which corresponds to the objective row. The first two columns contain:

NUMBER	Number of slack variable
ROW	Name of row

If $\tilde{u}_i = \tilde{l}_i$ for the slack variable under consideration, the remaining columns contain only the message FIXED VARIABLE.

In the case when the slack variable $\tilde{x}_i$ is nonbasic at its lower bound the message VARIABLE AT LOWER BOUND appears in the next two columns, which otherwise contain:

LL FOR L BOUND	Lower boundary of range B for $\tilde{l}_i$
UL FOR L BOUND	

The next two columns give similar information about the upper bound. In other words, if the slack variable $\tilde{x}_i$ is nonbasic at its upper bound, the message VARIABLE AT UPPER BOUND is printed; if not the columns contain:

LL FOR U BOUND	Lower boundary of range B for $\tilde{u}_i$
UL FOR U BOUND	Upper boundary of range B for $\tilde{u}_i$

The columns which follow all appear under the heading CHANGES AT BDRIES (OR OPT SOL VANISHES). These are used only for nonbasic slack variables, remaining blank for basic variables.

If the slack variable $\tilde{x}_i$ is at its lower bound then the columns contain the names of the basic variables which change their state at the boundaries of range B for $\tilde{l}_i$, given that the solution does not vanish. The message LL indicates that the variable has reached its lower bound, while UL shows that the upper bound has been reached. The first column, headed LOWER, gives the name of the variable which changes its state at the lower boundary of range B for $\tilde{u}_i$; the second column, headed UPPER, gives the name of the variable which changes its state at the upper boundary. The name of $\tilde{x}_i$ may also appear under the heading UPPER. This means that $\tilde{u}_i$ is the upper boundary of range B for $\tilde{l}_i$ and the set of feasible solutions is then empty beyond the boundary.

If the slack variable $\tilde{x}_i$ is at its upper bound, these columns contain the names of the basic variables which change their states at the boundaries of range B for $\tilde{u}_i$, given that the solution does not vanish. The messages LL and UL have the same meaning as above. The first column, headed LOWER, gives the name of the variable which changes its state at the lower boundary of range B for $\tilde{u}_i$; the second column, headed UPPER, gives the name of the variable which changes its state at the upper boundary. If the name of $\tilde{x}_i$ appears under the heading LOWER, then $\tilde{l}_i$ is the lower boundary of range B for $\tilde{u}_i$ and the set of feasible solutions is then empty beyond this boundary.

The last column contains:

I	Row number

SECTION 2 - COLUMNS contains information analogous to that described above for each structural variable $\tilde{x}_i$, $i = 1,...,n$. All of the information may be interpreted in the same way as in SECTION 1 - ROWS, with the following exceptions:

NUMBER	Number of structural variable
COLUMN	Name of structural variable
M+J	$m+1+i$

Beware: in most cases BRAN does not know if there is an optimal solution beyond the boundaries of range B and so a variable name may be printed under

CHANGES AT BDRIES (OR OPT SOL VANISHES) even if the optimal solution vanishes. The question whether the optimal solution exists may be answered (in the negative) only if the name of the nonbasic variable appears in the appropriate column of the output.

1.3.4. ELMRAN

ELMRAN performs ordinary ranging on the elements of the constraint matrix col($a_1, a_2, \ldots, a_m$) (see (2)). For selected elements a_j^i, $i = 1,\ldots,n, j = 1,\ldots,m$ the subroutine determines the largest open range in which a_j^i may vary without affecting the optimal basis or the state of nonbasic variables. The list of the selected elements is given in the data. While the range for a_j^i is being determined, all other elements a_l^k, k different from i and/or j different from l, are fixed at their original values. ELMRAN also gives the sensitivity of the optimal cost with respect to the elements. In addition, some information on the change of state of variables at the boundaries is given.

Data: see Section 1.2.

The output is entitled MATRIX ELEMENT RANGING. The following information is then printed for each of the selected elements a_j^i:

NUMBER	Number of slack variable
ROW	Name of row
NUMBER	Number of column
COLUMN	Name of column
ELEM VALUE	Value of the selected element of the constraint matrix
LOWER LIMIT	Lower boundary of the range in which the element may vary without affecting the optimal basis or the state of nonbasic variables
UPPER LIMIT	Upper boundary of this range
SENSITIVITY	Sensitivity of optimal cost with respect to the selected element
CHANGE AT LL	Name of the variable which changes its state at the lower boundary. No name is printed if the boundary is infinite or the optimal basis becomes singular at the boundary. TO LL is printed if this variable becomes nonbasic at its lower bound, OFF LL is printed if it is nonbasic and leaves its lower bound. Similarly, TO UL and OFF UL are printed for the upper bound. (Beware: ELMRAN does not know if there is an optimal solution beyond the boundary and so a variable name may be printed even if the optimal solution vanishes)
CHANGE AT UL	Name of the variable which changes its state at the upper boundary. Other explanations as above.

If the row number is the number of objective row, the following message is displayed:

XXX THE REQUIRED NUMBER OF ROW IS THE NUMBER OF OBJECTIVE ROW

and no analysis is performed for this particular element.

1.3.5. DIRRAN

DIRRAN performs directional ranging on the costs. For a given increment $\Delta a_0 \in R_n$ of the cost vector a_0, this subroutine determines the largest real $t_{\max} > 0$ such that for every cost vector of the form $a_0 + t\Delta a_0$, $t \in [0, t_{\max}]$, the optimal solution is the same as at the point a_0 (i.e., at $t = 0$). The boundary cost components $a_0^i + t\max\Delta a_0^i$, $i = 1,\ldots,n$, and some information on the change of state of variables at the boundary is also given. Beware: if a structural variable, say $\tilde{x}_i$, is fixed, then Δa_0^i is automatically set to zero, regardless of the value given in the data.

Data: see Section 1.2.

The output is entitled DIRECTIONAL COST RANGING. It takes one of two forms, depending on the value of $t_{\max}$. If $t_{\max} < 10^{15}$, we have the finite range case, while if $t_{\max} > 10^{15}$ we have the infinite range case. Let us consider the finite range first.

In this case the sub-heading FINITE RANGE is printed below the main title, with the corresponding value of $t_{\max}$ in brackets. Next, the following information is given for each structural variable $\tilde{x}_i$, $i = 1,...,n$:

NUMBER	Number of structural variable
COLUMN	Name of structural variable
DIRECTION	Increment component Δa_0^i
OBJ GRADIENT	Cost component a_0^i
BOUNDARY VALUE	Boundary value of cost component $(a_0^i + t_{\max}\Delta a_0^i)$
M+J	$m+1+i$

At the boundary $t{=}t_{\max}$ either the optimal solution vanishes or one of the nonbasic variables changes its state. The name and original state of this variable are given in the last row of the output in the form: AT BOUNDARY VARIABLE "name" CEASES TO BE AT "bound" OR OPTIMAL SOLUTION VANISHES. The letters LL are substituted for "bound" if the variable is no longer at its lower bound, while UL appears if the variable is no longer at its upper bound.

In the infinite range case $(t_{\max} \geq 10^{15})$ the message INFINITE RANGE (TMAX.GE.1.E15) is displayed. Beneath this the same information is given for each structural variable as in the finite range case, except for the BOUNDARY VALUE, which is no longer relevant.

1.3.6. DRHSRN

DRHSRN performs directional ranging on the right-hand sides. For a given vector of increments $\Delta\tilde{b} \in R^{m+1}$ of the rhs vector $\tilde{b}$, DRHSRN determines the largest real $t_{\max} \geq 0$ such that for every rhs of the form $\tilde{b} + t\Delta\tilde{b}$, $t \in [0, t_{\max}]$, the optimal basis is the same as at the point $\tilde{b}$ (i.e., at $t = 0$). $\Delta\tilde{b}_{\text{obj}}$ is automatically set to zero.

Data: see Section 1.2.

The output is entitled DIRECTIONAL RHS RANGING. It takes one of two forms, depending on the value of $t_{\max}$. If $t_{\max} < 10^{15}$, we have the finite range case, while if $t_{\max} \geq 10^{15}$ we have the infinite range case. Let us consider the finite range first.

In this case the sub-heading FINITE RANGE is printed below the main title, with the corresponding value of $t_{\max}$ in brackets. Next, the following information is given for each row (or each slack variable $\tilde{x}_i, i = n+2,...,n+m+2$), except for the objective row (or the slack variable $\tilde{x}_{n+1+\text{obj}}$):

NUMBER	Number of slack variable
ROW	Name of row
DIRECTION	Component $\Delta\tilde{b}_i$ of the increment vector
RHS	Right-hand side component $\tilde{b}_i$
BOUNDARY VALUE	Boundary value of the rhs component $(\tilde{b}_i + t_{\max}\Delta\tilde{b}_i)$
I	Row number

At the boundary $t = t_{\max}$ either the optimal solution vanishes or one of the basic variables changes its state. The name and type of change are given in the last row of the output in the form: AT THE BOUNDARY VARIABLE "name" PASSES FROM THE BASIS TO "bound" OR OPTIMAL SOLUTION VANISHES. The letters LL are substituted for "bound" if the variable reaches its lower bound and UL if it reaches its upper bound.

In the infinite range case $(t_{\max} \geq 10^{15})$ the message INFINITE RANGE (TMAX.GE.1.E15) is displayed. Beneath this the same information is given for each non-objective row as in the finite range case, except for the BOUNDARY VALUE, which is no longer relevant.

1.3.7. DBRAN

DBRAN performs directional ranging on the bounds. For a given vector of increments col $(\Delta\tilde{l},\Delta\tilde{u}) \in R^{2(n+m+2)}$ of the vector of bounds col $(\tilde{l},\tilde{u})$, this subroutine determines two real numbers:

- $t_{\text{maxa}} \geq 0$, the largest real number such that for every bound vector of the form col $(\tilde{l},\tilde{u})$ + t col $(\Delta\tilde{l},\Delta\tilde{u})$, $t \in [0,t_{\text{maxa}}]$ the optimal solution is the same as for the bound vector col $(\tilde{l},\tilde{u})$, i.e., at $t = 0$.
- $t_{\text{maxb}} \geq 0$, the largest real number such that for every bound vector of the form col $(\tilde{l},\tilde{u})$ + t col $(\Delta\tilde{l},\Delta\tilde{u})$, $t \in [0,t_{\text{maxb}}]$ the optimal basis is the same as for the bound vector col $(\tilde{l},\tilde{u})$, i.e., at $t = 0$.

The bound increments $\Delta\tilde{l}_i,\Delta\tilde{u}_i$ which correspond to fixed variables are automatically set to zero regardless of the values given in the data.

Data: see Section 1.2.

The output is entitled DIRECTIONAL BOUND RANGING. Information on t_{maxa} is given under the heading A.NO CHANGE IN OPTIMAL SOLUTION. If $t_{\text{maxa}} < 10^{15}$ the message FINITE RANGE is displayed, with the corresponding value of t_{maxa} in brackets. If $t_{\text{maxa}} \geq 10^{15}$, INFINITE RANGE (TMAXA.GE.1.E15) is printed. Similar information on t_{maxb} is given under heading B.NO CHANGE IN THE OPTIMAL BASIS. The rest of the output is divided into two sections: SECTION 1 - ROWS and SECTION 2 - COLUMNS.

SECTION 1 - ROWS contains the following information for each slack variable $\tilde{x}_i$, $i = n+2,...,n+m+2$ (or for each row), except for the slack variable $\tilde{x}_{n+1+\text{obj}}$ which corresponds to the objective row:

NUMBER	Number of slack variable
ROW	Name of row
LL DIRECTION	Component $\Delta\tilde{l}_i$ of the lower bound increment vector $\Delta\tilde{l}$
LL BOUNDARY A	Boundary value of the lower bound $\tilde{l}_i + t_{\text{maxa}}\Delta\tilde{l}_i$; this is printed only if $t_{\text{maxa}} < 10^{15}$
LL BOUNDARY B	Boundary value of the lower bound $\tilde{l}_i + t_{\text{maxb}}\Delta\tilde{l}_i$; this is printed only if $t_{\text{maxb}} < 10^{15}$
UL DIRECTION	Component $\Delta\tilde{u}_i$ of the upper bound increment vector $\Delta\tilde{u}$
UL BOUNDARY A	Boundary value of the upper bound $\tilde{u}_i + t_{\text{maxa}}\Delta\tilde{u}_i$; this is printed only if $t_{\text{maxa}} < 10^{15}$
UL BOUNDARY B	Boundary value of the upper bound $\tilde{u}_i + t_{\text{maxb}}\Delta\tilde{u}_i$; this is printed only if $t_{\text{maxb}} < 10^{15}$
I	Row number

SECTION 2 - COLUMNS contains information analogous to that described above for each structural variable $\tilde{x}_i$, $i = 1,...,n$, with the following exceptions:

NUMBER	Number of structural variable
COLUMN	Name of structural variable
M+J	$m+1+i$

The last two rows of the output contain information on the change of state of variables at the boundaries. If $t_{\text{maxa}} < 10^{15}$, the message AT THE BOUNDARY A THE VARIABLE "name" HITS "bound" is displayed. The letters LL are substituted for "bound" if the variable hits its lower bound and letters UL if it hits the upper bound. If $t_{\text{maxb}} < 10^{15}$, the message AT THE BOUNDARY B THE BASIC VARIABLE "name" BECOMES

NONBASIC AT "bound" OR OPTIMAL SOLUTION VANISHES is displayed in the next row. Once again, LL is used to denote the lower bound and UL the upper bound.

1.3.8. ROWRAN

ROWRAN performs directional ranging on the rows of the constraint matrix. For a given vector of increments $\Delta a_i \in R^n$ of the constraint vector a_i, $i = 1,...,m$, ROWRAN determines the largest open range $(t_{\min}, t_{\max})$ such that for every i-th constraint vector of the form $a_i + t\Delta a_i$, $t \in (t_{\min}, t_{\max})$, the optimal basis and the state of nonbasic variables are the same as at the point a_i (i.e., at $t = 0$).

Data: see Section 1.2.

The output is entitled MATRIX ROW RANGING. The following information is given in the first two rows of the output:

NUMBER	Number of slack variable
ROW	Name of row
TMIN	Lower boundary of the range
TMAX	Upper boundary of the range
SENSITIVITY	Sensitivity of optimal cost with respect to parameter t
CHANGE AT LL	Name of the variable which changes its state at the lower boundary. No name is printed if the boundary is infinite or the optimal basis becomes singular at the boundary. TO LL is printed if this variable becomes nonbasic at its lower bound, OFF LL is printed if it is nonbasic and leaves its lower bound. Similarly, TO UL and OFF UL are printed for the upper bound. (Beware: ROWRAN does not know if there is an optimal solution beyond the boundary and so a variable name may be printed even if the optimal solution vanishes)
CHANGE AT UL	Name of variable which changes its state at the upper boundary. Other explanations as above
I	Number of row

Next, the following information is given for each structural variable:

NUMBER	Number of column
COLUMN	Name of column
DIRECTION	Increment component Δa_i^k
ELEM VALUE	Component a_i^k of the constraint vector a_i
LOWER LIMIT	Lower boundary value of the component of the constraint vector $(a_i^k + t_{\min}\Delta a_i^k)$; this is printed only if $t_{\min} > -10^{15}$
UPPER LIMIT	Upper boundary value $(a_i^k + t_{\max}\Delta a_i^k)$; only if $t_{\max} < 10^{15}$
M+J	$m+1+k$

1.3.9. COLRAN

COLRAN performs directional ranging on the structural columns of the constraint matrix. For a given vector of increments Δa^i of the column a^i, $i = 1,...,n$, COLRAN determines the largest range $(t_{\min}, t_{\max})$ such that for every i-th constraint column of the form $a^i + t\Delta a^i, t \in (t_{\min}, t_{\max})$ the optimal basis and the state of nonbasic variables are the same as at the point a^i (i.e., at $t = 0$). Δa^i_{obj} is automatically set to zero.

Data: see Section 1.2.

The output is entitled MATRIX COLUMN RANGING. The following information is given in the first two rows of the output:

NUMBER	Number of column
COLUMN	Name of column
TMIN	Lower boundary of the range
TMAX	Upper boundary of the range
SENSITIVITY	Sensitivity of optimal cost with respect to parameter t
CHANGE AT LL	Name of the variable which changes its state at the lower boundary. No name is printed if the boundary is infinite or the optimal basis becomes singular at the boundary. TO LL is printed if this variable becomes nonbasic at its lower bound, OFF LL is printed if it is nonbasic and leaves its lower bound. Similarly, TO UL and OFF UL are printed for the upper bound. (Beware: COLRAN does not know if there is an optimal solution beyond the boundary and so a variable name may be printed even if the optimal solution vanishes)
CHANGE AT UL	Name of the variable which changes its state at the upper boundary. Other explanations as above
M+J	$m+1+i$

Next, the following information is given for each row (except for the objective row):

NUMBER	Number of slack variable
ROW	Name of row
DIRECTION	Increment component Δa_k^i
ELEM VALUE	Component a_k^i of the constraint column a^i
LOWER LIMIT	Lower boundary value of the component of the constraint column $(a_k^i + t_{\min}\Delta a_k^i)$; this is printed only if $t_{\min} > -10^{15}$
UPPER LIMIT	Upper boundary value $(a_k^i + t_{\max}\Delta a_k^i)$; only if $t_{\max} < 10^{15}$
I	Number of row

1.4. Additional information about outputs

Errors are indicated by messages beginning with XXX. Fatal errors which occur when the Data Ranging File for a particular type of postoptimal analysis is being processed do not stop postoptimal analysis of other types. The outputs are printed in the following order: Directional Cost Ranging, Directional Bound Ranging, Directional RHS Ranging, Cost Ranging, Bound Ranging, RHS Ranging, Matrix Element Ranging, Matrix Row Ranging and Matrix Column Ranging.

NONE is printed in place of any number whose absolute value is greater than or equal to 10^{15} (such numbers are taken as equal to infinity). For greater clarity of the output, format F16.5 is used and so it may happen that a number is out of format. In this case, asterisks are printed.

1.5. An Example

We shall now illustrate the performance of POSTAN using a simple example. The linear programming problem is as follows: Maximize

$$F(x) = x_1 - x_2 + 0.5x_3 + 2x_4 + 3x_5$$

subject to

$$1.1x_1 + 1.2x_2 + 1.3x_3 \leq 7$$

$$0.1x_3 + 0.2x_4 - x_5 \geq -7$$

$$-10.6 \leq 2.1x_3 + 2.2x_4 \leq 10.7$$

$$x_1 + x_2 + x_3 = 0.01$$

$$x_1 \leq 1.5 \ , \quad x_2 \geq -1.4 \ , \quad 0 \leq x_3 \leq 10 \ , \quad x_4 = 2 \quad .$$

An additional constraint is introduced as the fourth row:

$$-\infty < 5x_1 + 5x_2 + 5x_3 + 5x_4 < +\infty$$

in order to show the effect of a free constraint on the POSTAN output.

Below, exemplary of output of the subroutines DIRRAN, DBRAN, DRHSRN, CRAN, BRAN, RHSRAN, ELMRAN, ROWRAN and COLRAN is reproduced.

```
name          test
rows
 n  ob
 l  r1
 g  r2
 l  r3
 e  r4
 n  de
columns
    x1        ob            1.
    x1        r1            1.1
    x1        r4            1.
    x1        de            5.
    x2        ob           -1.
    x2        r1            1.2
    x2        r4            1.
    x2        de            5.
    x3        ob             .5
    x3        r1            1.3
    x3        r2             .1
    x3        r3            2.1
    x3        r4            1.
    x3        de            5.
    x4        ob            2.
    x4        r2             .2
    x4        r3            2.2
    x4        de            5.
    x5        ob            3.
    x5        r2           -1.
rhs
    rh        r1            7.
    rh        r2           -7.
    rh        r3           10.7
    rh        r4           .01
ranges
    ra        r3          -21.3
bounds
 mi bo        x1
 up bo        x1            1.5
 lo bo        x2           -1.4
 up bo        x3           10.
 fx bo        x4            2.
 fr bo        x5
endata


             m i n o s     ---    version  4.0     mar  1981
             = = = = =

specs file
----------
           begin POSTAN Example for Linear Case
                 maximize
                 objective ob
                 data ranging file 9
                 print data ranging file
                 directional cost ranging
                 directional bounds ranging
                 directional rhs ranging
                 by-elements cost ranging
                 by-elements bounds ranging
                 by-elements rhs ranging
                 matrix element ranging
                 matrix row ranging
                 matrix column ranging
           end

              P O S T A N    ---    Postoptimal Analysis Package , version  3.0    Oct    1987
              ===========


data ranging file
-----------------
       1  name          dcos
       2  'set'                     0.100000d+01
       3  *
       4  *    Empty set of data for directional cost ranging.
       5  *
xxx empty ranging file. all variables are set to 0.100000d+01
       6  endata


directional cost ranging
------------------------

finite range (tmax=   0.20000d+01)

 number  .column.  ...direction..  .obj.gradient.  boundary value   m+j

      1  x1               1.00000         1.00000         3.00000     7
      2  x2               1.00000        -1.00000         1.00000     8
      3  x3               1.00000          .50000         2.50000     9
      4  x4                .00000         2.00000         2.00000    10
      5  x5               1.00000         3.00000         5.00000    11

at boundary variable x3        ceases to be at 11 or optimal solution vanishes


data ranging file
-----------------
       8  name          dbou
       9  'set'                     0.100000d+01
      10      lower     x4             .00000d+00                .00000d+00
      11      lower     r1             .00000d+00                .00000d+00
      12      lower     r2             .00000d+00                .00000d+00
      13                r3             .00000d+00                .00000d+00
      14      lower     r4             .00000d+00                .00000d+00
      15      lower     de          .00000d+00               .00000d+00
      16      upper     x4          .00000d+00   r1          .00000d+00
      17                r2          .00000d+00               .00000d+00
      18                de          .00000d+00   r4          .00000d+00
      19   endata


directional bound ranging
-------------------------

a. no change in the optimal solution
   finite range (tmaxa=     .00000d+00)

b. no change in the optimal basis
   finite range (tmaxb=   0.30000d+01)

section 1 - rows

 number  ...row..  .ll direction.  .ll boundary a  .ll boundary b  .ul direction.  .ul boundary a  .ul boundary b   ..i

      8  r1               .00000          .00000          .00000          .00000            none            none    2
      9  r2               .00000            none            none          .00000          .00000          .00000    3
     10  r3               .00000          .00000          .00000         1.00000        21.30000        24.30000    4
     11  r4               .00000          .00000          .00000          .00000          .00000          .00000    5
     12  de               .00000            none            none          .00000            none            none    6

section 2 - columns

 number  .column.  .ll direction.  .ll boundary a  .ll boundary b  .ul direction.  .ul boundary a  .ul boundary b   m+j

      1  x1              1.00000            none            none         1.00000         1.50000         4.50000    7
      2  x2              1.00000        -1.40000         1.60000         1.00000            none            none    8
      3  x3              1.00000          .00000         3.00000         1.00000        10.00000        13.00000    9
      4  x4               .00000         2.00000         2.00000          .00000         2.00000         2.00000   10
      5  x5              1.00000            none            none         1.00000            none            none   11

at the boundary a the variable x2       hits ll

at the boundary b the basic variable  r3       becomes nonbasic at ll or optimal solution vanishes


data ranging file
-----------------
      20  name          drhs
      21  'set'                    0.100000d+01
      22                r4          .00000d+00               .00000d+00
      23  endata


directional rhs ranging
-----------------------

finite range (tmax=0.15000d+02)

 number  ...row..  ...direction..  .........rhs..  boundary value   ..i

      8  r1              1.00000         7.00000        22.00000     2
      9  r2              1.00000        -7.00000         8.00000     3
     10  r3              1.00000        10.70000        25.70000     4
     11  r4               .00000          .01000          .01000     5
     12  de              1.00000          .00000        15.00000     6

at the boundary variabls r3       passes from the basis to ul or optimal solution vanishes
```

```
cost ranging - linear case
------------

 number  .column.  .obj gradient.  ..lower limit.  ..upper limit.  .change at lower limit.  .change at upper limit.  m+j
                                                                    or opt sol vanishes      or opt sol vanishes

      1  x1          1.00000         .800000            none          x3                                            7
      2  x2         -1.00000              none     1.00000                                     x2                   8
      3  x3          .500000              none     .700000                                     x3                   9
      4  x4           2.0000          2.0000       2.0000                                                          10
      5  x5          3.00000              0.       5.00000            r2                       x3                  11


bound ranging
-------------

a. no solution change

section 1 - rows

 number  ...row..  .ll for l bound.  .ul for l bound.  .ll for u bound.  .ul for u bound.  ..i

      8  r1                   none           7.12900           7.12900              none    2
      9  r2                   none            .00000      variable at upper bound           3
     10  r3                   none           6.30000           6.30000              none    4
     11  r4         fixed variable                                                          5
     12  de                   none         -10.05000         -10.05000              none    6
 section 2 - columns

 number  .column.  .ll for l bound.  .ul for l bound.  .ll for u bound.  .ul for u bound.  m+j

      1  x1                   none           1.41000           1.41000              none    7
      2  x2           variable at lower bound                 -1.40000              none    8
      3  x3           variable at lower bound                   .00000              none    9
      4  x4         fixed variable                                                         10
      5  x5                   none           7.40000           7.40000              none   11

b. no basis change

section 1 - rows

 number  ...row..  .ll for l bound.  .ul for l bound.  .ll for u bound.  .ul for u bound  ...changes at bdries...   ..i
                                                                                          (or opt sol vanishes)
                                                                                           lower        upper

      8  r1                   none           7.12900           7.12900              none                             2
      9  r2                   none            .00000              none              none                             3
     10  r3                   none           6.30000           6.30000              none                             4
     11  r4         fixed variable                                                                                   5
     12  de                   none         -10.05000         -10.05000              none                             6

 section 2 - columns

 number  .column.  .ll for l bound.  .ul for l bound.  .ll for u bound.  .ul for u bound  ...changes at bdries...   m+j
                                                                                          (or opt sol vanishes)
                                                                                           lower        upper

      1  x1                   none           1.41000           1.41000              none                             7
      2  x2               -1.49000          69.88998          -1.40000              none  x1     ul   r1      ll     8
      3  x3                -.09000           3.00000            .00000              none  x1     ul   r3      ll     9
      4  x4         fixed variable                                                                                  10
      5  x5                   none           7.40000           7.40000              none                            11


rhs ranging
```

```
-----------

 number  ...row..  .....rhs......  ..lower limit.  ..upper limit.  .change at lower limit.  .change at upper limit.   m+j
                                                                   (or opt sol vanishes)    (or opt sol vanishes)

      8  r1              7.00000         -.12900           none       r1       ll                                    2
      9  r2             -7.00000            none           none                                                      3
     10  r3             10.70000         4.40000       25.70000       r3       ll            r3       ul             4
     11  r4               .01000            none         .10000                              x1       ul             5
     12  de               .00000            none           none                                                      6

data ranging file
-----------------
     24  name          mele
     25      x3        r1        .00000d+00  r2        .00000d+00
     26                r3        .00000d+00  r4        .00000d+00
     27                de        .00000d+00            .00000d+00
     28  endata


matrix element ranging
----------------------

 number  ...row..  number  .column.  .elem value.  lower limit.  upper limit.  sensitivity.  .change at ll.  .change at ul.

      8  r1             3  x3           1.30000          none          none    .0000d+00
      9  r2             3  x3            .10000          none        .16667    .0000d+00                    x3      off ll
     10  r3             3  x3           2.10000          none          none    .0000d+00
     11  r4             3  x3           1.00000        .80000          none    .0000d+00  x3      off ll
     12  de             3  x3           5.00000          none          none    .0000d+00


data ranging file
-----------------
     29  name          mrow
     30  'set'                    0.100000d+01
     31      r1        x3         .00000d+00            .00000d+00
     32  endata


matrix row ranging
------------------

 number  ...row..  .....tmin.....  .....tmax.....  ..sensitivity.  ..change at ll..  ..change at ul..     i

      8  r1                  none          .75760          .00000                       r1       to ll     2

 number  .column.  ...direction..  ..elem value..  ..lower limit.  ..upper limit.  m+j

      1  x1               1.00000         1.10000            none         1.85760    7
      2  x2               1.00000         1.20000            none         1.95760    8
      3  x3                .00000         1.30000         1.30000         1.30000    9
      4  x4               1.00000          .00000            none          .75760   10
      5  x5               1.00000          .00000            none          .75760   11


data ranging file
-----------------
     33  name          mcol
     34  'set'                    0.100000d+01
     35      x3        r1         .00000d+00            .00000d+00
     36  endata


matrix column ranging
```

```
---------------------

number  .column.  .....tmin.....  .....tmax.....  ..sensitivity.  ..change at ll..  ..change at ul..  m+j

   3  x3               none          .10000          .00000                    x3        off ll     9

number  ...row..  ...direction..  ..elem value..  ..lower limit.  ..upper limit.    i

   8  r1             .00000        1.30000         1.30000         1.30000      2
   9  r2            1.00000         .10000           none           .20000      3
  10  r3            1.00000        2.10000           none          2.20000      4
  11  r4            1.00000        1.00000           none          1.10000      5
  12  de            1.00000        5.00000           none          5.10000      6
```

2. IMPLEMENTATION OF LINEAR - FRACTIONAL POSTAN IN MINOS

In order to insert the POSTAN LFP into MINOS, and to allow it to be used in the same way as other MINOS facilities, the following changes have been made.

2.1. Preparing LFP problem for solving

The LFP problem (105)-(107) is prepared in MPS standard and placed in MINOS MPS file. The rows of numerator and denominator of the objective function ought to be of type N (free).

The LFP mode is initiated when the following keyword is found in SPECS file :

DENOMINATOR (name of row)

There is also a declaration of the name of the row in MPS file which will be interpreted as the denominator of objective function. If the of name the row declared as above is not found in MPS file the following warning message:

NO OBJECTIVE'S DENOMINATOR FOUND

is printed and the package returns to LP mode of solving. The row that is the numerator of objective function is declared in SPECS file in the same way as the row of objective function in LP mode, i.e. either by the keyword

OBJECTIVE (name of row)

or in the case when the above word does not occur first row of N type will be regarded as the numerator.

The LFP mode cooperates with the linear part of MINOS package and so all keywords concerning that part maintain their function in LFP mode. Using the keywords concerning the nonlinear part of MINOS may cause incorrect work of the package.

The new stopping condition of the simplex algorithm arises. If the denominator occurs to be negative the package quits its run with a status ***unbounded solution.*** The iteration number and the value of denominator is printed.

The form of information given by the package during its work and the form of final results is clear and requires no additional comments. The only exception here is DUAL ACTIVITY column in the printout. In LFP mode it contains the value computed from (104) after substitution in place of the relative costs directly the derivative computed for the optimal solution.

2.2. Changes in MINOS package procedures.

All changes in the package procedures are done by inserting groups of instructions performed only in LFP mode. Two global variables have been included located in COMMON block called COMGG and several local work variables in procedures. The following procedures have been extended:

SPECS 2	recognition of the keyword initiating LFP mode,
MPS	recognition of the numerator and denominator of the objective function in input data,
FORMC	computing the modified relative cost,
DRIVER, ITEROP	proper monitoring of calculation in LFP mode,
SOLN	output of LFP results.

2.3. Cost ranging subroutine LFP CRAN

Ranging of cost in LFP mode is implemented as an alternative procedure to the original CRAN. Since POSTAN 3 always knows the mode of its operation it calls the proper variant of the subroutine as an answer to the same keyword in SPECS file.

LFP CRAN performs ordinary ranging on the costs. For each cost component of the numerator c_0^i, $i = 1,...,n$ and the denominator d_0^i, $i = 1,...,n$, the subroutine determines the largest range in which they may vary without affecting the optimal solution. While the range for c_0^i is being determined, all other components c_0^j, $j \neq i$ and d_0, remain fixed at their original values. The similar assumption is held when a component of denominator is analyzed. LFP CRAN also gives some information on the change of state of variables at the boundaries.

This subroutine does not require any input data.

The output is entitled COST RANGING. The following information is then given for each cost component of the numerator and the denominator, $i = 1,...,n$:

NUMBER	Number of structural variable
COLUMN	Name of structural variable
OBJ GRADIENT	Cost component
LOWER LIMIT	Lower boundary of the range in which the cost component may vary without affecting the optimal solution
UPPER LIMIT	Upper boundary of this range
CHANGE AT LOWER LIMIT (OR OPTSOL VANISHES)	Name of the nonbasic variable which changes its state at the lower boundary; this is printed only if the lower boundary is finite. (Beware: CRAN does not know if there is an optimal solution beyond the boundary so that the name of a nonbasic variable may be printed even if the optimal solution vanishes)
CHANGE AT UPPER LIMIT (OR OPTSOL VANISHES)	Name of the nonbasic variable which changes its state at the upper boundary (other explanations as above)
M+J	NUMBER $+ m + 1$

To improve readability of the printout a line about c_0^i precedes the line about d_0^i.

2.4. An Example

LFP option of POSTAN will be illustrated using the same programming problem as previously (see point 1.5) with the free row declared as a denominator of objective. An output of POSTAN for the case is reproduced below together with results of the by-elements cost ranging.

```
          m i n o s    ---    version 4.0   mar  1981
          = = = = =


specs file
----------
         begin POSTAN Example of Linear-Fractional Case
              maximize
              objective ob
              denominator de
         *
         *    Euations are the same as in the example for LP case.
         *    Caution:
         *    The denominator must be positive inside a feasible region!
         *
              by-elements cost ranging
         end

problem name    test                objective value    2.8865671615d+00

status          optimal soln        iteration   1     superbasics    0

objective       ob      /de        (max)
rhs             rh
ranges          ra
bounds          bo

section 1 - rows

 number  ...row..  at  ...activity...  slack activity  ..lower limit.  ..upper limit.  .dual activity   ..i

      7  ob        bs      29.010        -29.010           none            none       0.99502d-01      1
      8  r1        bs      -.12900        7.1290           none          7.0000            0.          2
      9  r2        ll      -7.0000         0.             -7.0000           none         .29851        3
     10  r3        bs       4.4000        6.3000         -10.600          10.700       0.41874d-16     4
     11  r4        eq     0.10000d-01      0.           0.10000e-01     0.10000e-01      1.3366        5
     12  de        bs       10.050       -10.050           none            none         -.28722        6

section 2 - columns

 number  .column.  at  ...activity...  .obj gradient.  ..lower limit.  ..upper limit.  reduced gradnt   m+j

      1  x1        bs       1.4100       -1.3366           none          1.5000             0.         7
      2  x2        ll      -1.4000       -1.5356         -1.4000            none          -.19900      8
      3  x3        ll         0.         -1.3864            0.            10.000       -0.19900d-01    9
      4  x4        eq       2.0000       -1.2371          2.0000          2.0000         -1.1774      10
      5  x5        bs       7.4000        .29851           none            none             0.        11
      6  rh        eq      -1.0000          0.            -1.0000         -1.0000         -2.0762      12


          P O S T A N    ---    Postoptimal Analysis Package , version  3.0    Oct    1987
          ===========


cost ranging - fractional case
```

```
------------

number  .column.  .obj gradient.  ..lower limit.  ..upper limit.  .change at lower limit.  .change at upper limit.  m+j
                                                                   or opt sol vanishes      or opt sol vanishes

     1  x1          1.00000         .800000              none       x3                                              7
     1  x1          5.00000       -2.12766        5.06997          de                       x3                      7
     2  x2         -1.00000              none     1.00000                                   x2                      8
     2  x2          5.00000        4.23312       12.1786           x2                       x2                      8
     3  x3          .500000              none     2.50000                                   x3                      9
     3  x3          5.00000        4.30714               none       x3                                              9
     4  x4           2.0000         2.0000         2.0000                                                          10
     4  x4           5.0000         5.0000         5.0000                                                          10
     5  x5          3.00000             0.        5.00000          de                       x3                     11
     5  x5               0.       -.458799        4.42731          x3                       de                     11
```

3. AUXILIARY ROUTINES IN POSTAN 3

3.1. Programming a sequence of optimization problems in MINOS

3.1.1. General information.

The MODIF module is an extended version of the user's interface for the linear and non-linear programming package MINOS.

MODIF controls performance of a sequence of optimization problems arising as successive modifications of the basic problem recorded with the use of MPSX format on MPS file of MINOS.

The module's work is controlled by the contents of MODIFICATION file which can be interpreted as a series of "exchange" type alterations concerning the MPS file. In fact the contents of package arrays containing the LP problem matrix are changed which eliminates the time-consuming operation of MPS file processing.

The module restarts automatically (in "flying" manner) the SIMPLEX algorithm assuming the basis of the optimal solution to the last problem in the series. Due to this the realization time is considerably shortened and the user need not manipulate the files containing the basis images.

Furthermore the module enables the user to change optimization criteria for the linear and linear-fractional case by indicating the rows of MPS file which become a new objective function or its numerator and denominator.

This description will be regarded further on as development of MINOS documentation [2]. The reader is assumed to be acquainted also with the MPSX standard. All the references to those positions are omitted in the text.

3.1.2. Data processed by the module

MODIF module is initialized on user's demand by the MINOS package control.

The following data sets are used during the module's work:

- initializing data located in SPECS file,
- control data located in MODIFICATION file,
- data concerning the problem located in package arrays.

3.1.3. Initialization of the module.

The module is initialized after placing new keywords in the control file of MINOS called SPECS file.

Key	Default	Meaning
MPS MODIFICA-TION	off	The modification module is activated.
MODIFICA-TION FILE	as for MPS file	Internal file number of MINOS is assigned to the file that controls the module run.

In the case of module initialization the meaning of all other MINOS keywords remains the same but in respective cases action forced by using those instructions will be performed before the first or after the last course of the series (a series will be seen as a singular problem).

3.1.4. Programming of series of runs - MODIFICATION file.

MODIFICATION file is a character set built of records (lines) containing at most 80 characters. It may be divided into a few sections similar to MPSX standard. The record structure almost always accords with that standard. The extensions will be discussed below.

NAME section

Must occur as the first declaration in MODIFICATION file. The module checks consistence of the given name with the name of the basic problem (MPS file).

ROWS section

The changes of rows types (the way of their limitation) are described here. Values of right hand sides of modified rows remain unchanged.

COLUMNS section

The section introduces the changes of values of matrix elements existing in the basic problem. "Existing" means here not rejected by the package when loading MPS file.

RHS section

The section introduces changes of values of right-hand sides existing in the basic problem. "Existing" means here non-zero i.e. declared explicitly in MPS file.

RANGES section

The section introduces the opposite side constraints for the rows declared in MPS file. The above action is performed for each row of L,G,E type regardless if the constraint was introduced or not in the basic problem.

BOUNDS section

The section introduces the constraints for the structural variables of the model.

OBJECTIVE section

This is an extension to MPSX standard. It consists of one record having the following structure :

positions 1-9	- "OBJECTIVE",
positions 11-18	- the name of row in the basic problem that is to become the row of an objective function,
positions 21-32	- "1" in the case of minimization, "-1" in the case of maximization.

The row declared in this section automatically becomes of the N type (its right-hand-side is deleted) in the case it was declared a non-zero one before.

DENOMINAT section

This is an extension to MPSX standard. It consists of one record of the following structure:

positions 1-9	- "DENOMINAT",
positions 11-18	- name of the row in basic problem that is to become the denominator of the objective function.

The row declared in this section becomes automatically of N type and its right-hand-side is deleted if it was declared as a non-zero one before. This declaration of the section causes an automatic switch of the package into the mode of solving the linear-fractional programming problems. The numerator of the objective function is the row defined either in SPECS file or as a result of using OBJECTIVE section.

NODENOMIN section

This is an extension of MPSX standard.

positions 1-9	- "NODENOMIN"

Declaration of this section causes switching the package into the mode of solving linear programming problem with an objective function declared before as a numerator. The row of denominator remains of the N type.

ENDATA section

The appearance of this section finishes the modification of the problem. The SIMPLEX algorithm is re-started automatically. By placing this section just after the NAME section that opens the sequence, the computations of the non-modified basic problem (MPS file) is caused.

General remarks for MPS file construction

1. A series of problems being solved is constructed following the rule "modification to modification".

2. A modification for a consecutive problem in the series is declared using any sequence of sections excluding NAME and ENDATA ended by ENDATA section.
3. The modifications are recorded successively in the package arrays according to their order of appearance in MODIFICATION file. Their number is practically unlimited for one problem.
4. For RHS, RANGES and BOUNDS sections the module checks the names appearing in them accepting only those which accord with declarations in SPECS file.
5. The package remembers the type of the last section opened in the series (from among ROWS, COLUMNS, RHS, RANGES, BOUNDS). Therefore if the series of problems consists in repeated modification of the same coefficient it is not necessary to repeat the respective section marker. If the first modification of the series is not preceded by any of the markers mentioned above it is assumed that the RHS section has been opened.
6. Only 4 first characters of a marker are analyzed by the module.
7. Restrictions to the modification introduced by COLUMNS and RHS sections are forced by the structure of MINOS package arrays. If it is necessary, this inconvenience may be evaded in two ways:
   - by including to the basic problem (MPS file) coefficients or right-hand-side values e.g. equal to 1, which will be deleted during the first modification and in consecutive problems fixed on desired values,
   - by using PHANTOM mechanisms (see (3))
8. MODIFICATION file may contain several series of modifications for different basis problems starting with the NAME section with an appropriate name.

3.1.5. Emergency situations in module processing

Apart from the standard printout of MINOS, under the heading:

MODIFICATION file FOR CYCLE No n

appropriate sections of MODIFICATION file are reprinted by the module as well as information about errors in case they occurred ("n" is the consecutive number of the problem in the series).

Due to the rule of series construction: "modification to modification" most of errors are of fatal type and cause the current and the following problems to be desisted.

- in MODIFICATION file no NAME section was found with the name corresponding to the name of the base problem - no modification reply,
- no optimal solution exists for the preceding problem,
- formal incorrectness occurs in MODIFICATION file,
- modifications concerning the admissible area (RHS, RANGES and BOUNDS sections) reduce it to an empty set,
- any of names concerning the basic problem was not identified.

The module terminates its work correctly after having met the next NAME section or end of file marker just after the ENDATA section on the MODIFICATION file.

3.1.6. An Example

The option for programming a sequence of optimization problems of POSTAN will be illustrated using the same LFP programming problem as previously (see point 2.4). Successive modification to the basic problem are programmed to show optimal solutions if coefficients of x_1 variable of the numerator and denominator are assigned to the values from outside the ranges obtained as a result of the postoptimal analysis. An output of POSTAN for the case is reproduced below.

```
          m i n o s     ---     version  4.0    mar  1981
          = = = = =

specs file
----------
          begin POSTAN - Example of MPS Modification Option
               maximize
               objective ob
               denominator de
          *
          *    The basic problem is as in the previous example.
          *
               mps modification
               modification file 10
          end


modification file for cycle no.   2
-----------------------------------

      1  name          test
      2  columns                      .00000d+00               .00000d+00
      3     x1         ob          7.90000d-01                 .00000d+00
      4  *
      5  *   The above value makes the currant solution to be not optimal.
      6  *   See results of postopotimal analysis in the previous example.
      7  *

problem name    test                 objective value    2.8600895480d+00

status          optimal soln         iteration    1     superbasics    0

objective       ob      /de       (max)
rhs             rh
ranges          ra
bounds          bo

section 1 - rows

 number  ...row..  at  ...activity...  slack activity  ..lower limit.  ..upper limit.  .dual activity   ..i

      7  ob        bs      28.744         -28.744           none           none       0.99502d-01     1
      8  r1        bs      .47100          6.5290           none        7.0000           0.          2
      9  r2        ll     -7.0000           0.           -7.0000           none         .29851        3
     10  r3        ul      10.700           0.           -10.600        10.700      -0.47382d-03     4
     11  r4        eq    0.10000d-01        0.          0.10000e-01   0.10000e-01      1.3443        5
     12  de        bs      10.050         -10.050           none           none        -.28459        6

section 2 - columns

 number  .column.  at  ...activity...  .obj gradient.  ..lower limit.  ..upper limit.  reduced gradnt   m+j

      1  x1        bs     -1.5900         -1.3443           none        1.5000           0.          7
      2  x2        ll     -1.4000         -1.5224        -1.4000           none        -.17811        8
      3  x3        bs      3.0000         -1.3732          0.           10.000           0.          9
      4  x4        eq      2.0000         -1.2239         2.0000         2.0000        -1.1653       10
      5  x5        bs      7.7000          .29851           none           none          0.         11
      6  rh        eq     -1.0000           0.           -1.0000        -1.0000        -2.0812       12
```

```
modification file for cycle no.  3
-----------------------------------

      9      x1       ob        1.00000d+00                .00000d+00
     10      x1       de        5.07000d+00                .00000d+00
     11  *
     12  *   Similar test with respect to a denominator coefficient.
     13  *   The solution obtained is equal to the basic one.
     14  *

problem name   test                 objective value    2.8761537237d-01

status         optimal soln         iteration    0     superbasics    0

objective      ob      /de       (max)
rhs            rh
ranges         ra
bounds         bo

section 1 - rows

 number  ...row..  at  ...activity...  slack activity  ..lower limit.  ..upper limit.  .dual activity   ..i

      7  ob        bs     28.410         -28.410            none            none         .10062          1
      8  r1        bs      .47100         6.5290            none          7.0000          0.             2
      9  r2        ll     -7.0000         0.             -7.0000            none         .30185          3
     10  r3        ul     10.700          0.             -10.600          10.700       -0.46520d-05      4
     11  r4        eq     0.10000d-01     0.             0.10000e-01     0.10000e-01    1.3576           5
     12  de        bs      9.9387        -9.9387            none            none        -.28762          6

section 2 - columns

 number  .column.  at  ...activity...  .obj gradient.  ..lower limit.  ..upper limit.  reduced gradnt   m+j

      1  x1        bs     -1.5900        -1.3576            none          1.5000          0.             7
      2  x2        ll     -1.4000        -1.5387         -1.4000            none        -.18110          8
      3  x3        bs      3.0000        -1.3878          0.              10.000          0.             9
      4  x4        eq      2.0000        -1.2368          2.0000          2.0000        -1.1765          10
      5  x5        bs      7.7000         .30185            none            none          0.             11
      6  rh        eq     -1.0000         0.             -1.0000         -1.0000        -2.0994          12


modification file for cycle no.  4
-----------------------------------

     16      x1       de       -2.13000d+00                .00000d+00
     17  *
     18  *   The denominator becomes negative.
     19  *

itn    1 -- denominator is negative.  value =  -3.299990893d-03

problem name   test                 objective value   -8.7909333417d+03

status         unbounded            iteration    1     superbasics    0

objective      ob      /de       (max)
rhs            rh
ranges         ra
bounds         bo

section 1 - rows

 number  ...row..  at  ...activity...  slack activity  ..lower limit.  ..upper limit.  .dual activity   ..i
```

```
      7  ob          bs      29.010      -29.010            none          none   0.19639d+07    1
      8  r1          bs     -.12900       7.1290            none     7.0000             0.      2
      9  r2          ll     -7.0000           0.     -7.0000              none   0.58917d+07    3
     10  r3          bs      4.4000       6.3000     -10.600       10.700        0.90446d+07    4
     11  r4          eq     0.10000d-01       0.     0.10000e-01   0.10000e-01  -0.75207d+07    5
     12  de          bs    -0.33000d-02  0.33000d-02        none          none  -0.26088d+07    6

section 2 - columns

 number  .column.  at  ...activity...  .obj gradient.  ..lower limit.  ..upper limit.  reduced gradnt  m+j

      1  x1          bs      1.4100    0.56739d+07          none     1.5000       -0.65248d-09    7
      2  x2          ll     -1.4000   -0.13319d+08   -1.4000              none    -0.22529d+08    8
a     3  x3          ll          0.   -0.13320d+08        0.       10.000               0.      9
      4  x4          eq      2.0000   -0.13320d+08    2.0000       2.0000        0.11960d+08   10
      5  x5          bs      7.4000      -909.09            none          none           0.     11
      6  rh          eq     -1.0000           0.     -1.0000      -1.0000        0.55461d+08   12

xxx modification only when optimal solution
```

3.2. Module for decoding and selective output of results in MINOS

3.2.1. General information.

The REPORT module makes it possible to printout the LP solution according to user's demands. It makes the printout of MINOS easier and more legible for the user. The module is especially convenient when MINOS cooperates with MPS generator and a data base that usually contains description of LP model elements which may be used for reporting purposes.

The following functions are performed by the REPORT module:

- it gives the solution of LP problem described in natural language;
- it searches and outputs specified elements of the solution and formats them;
- it formats printed numbers by rescaling and rounding;
- it formats the whole printout;
- it makes simple computations on the elements of solution possible and presents their results.

In the discussed version the functions mentioned above concern the primary solution data.

The module uses data placed in the MINOS arrays therefore the printing of results proceeds without any intermediary file.

Required data are accessible through codes of rows and columns from MPS file.

It is assumed further on that this description is an extension of MINOS package documentation. Therefore all references to that documentation are omitted in the text.

3.2.2. Data processed by the module

The module is initialized on user's demand, by MINOS package control when an optimal solution of LP problem is obtained. The following data sets are used during the module's processing:

- data initializing the module placed in SPECS file,

- data concerning the solution placed in the package arrays,
- description and complementary data for printout placed in DICTIONARY file,
- printout stored in REPORT file,
- a file with decoded DICTIONARY file if the package performs a series of experiments or a user has demanded decoding of DICTIONARY file contents.

3.2.3. Module initialization.

There are a few new keywords that control the module run:

Key	Default	Meaning
REPORT YES	off	The REPORT module is activated.
REPORT FILE n	n = 6	Internal file number of MINOS is assigned to the file that the printout will be stored in.
DICTIONARY FILE n	n = 10	Internal file number of MINOS is assigned to the file that contains information controlling the report printing.
DECODE YES	NO	As recognition of codes used in DICTIONARY file is a time consuming process - there is a possibility of decoding DICTIONARY file especially recommended when REPORT module is to be used repeatedly for the same problem.

NOTE: Standard printout of the package is obtained independently from REPORT module action. Its output may be held up by a MINOS keyword:

SOLUTION NO

3.2.4. A printout programming - DICTIONARY file.

Both printout description data and complementary data are to be placed in DICTIONARY file arranged as specified below. The file is a set of 80 character records.

A printout file is created sequentially - one record of DICTIONARY file causes in principle only one consecutive line to be printed.

A record of DICTIONARY file contains 7 fields:

Field		Pos.	Format
1	Indicator	1-8	right aligned integer number or string of characters
2	Description	9-48	string of characters
3	Dimension	49-56	string of characters
4	Scale factor	57-63	real number
5	Format	64-71	string of characters
6	Omit indicator	72	character from among: space,1,2
7	Row code	73-80	string of characters

Field 1

Indicator field contents have two main functions:

A. When it is a MPS code (a string of characters) or a number greater than 0 it indicates the solution data that is to be printed out. A positive number is interpreted as a consecutive number of row and column of LP problem, that is established by the package in order of appearance in MPS file and printed in NUMBER column of the standard printout of the solution.

   NOTE: Changes in MPS file such as adding, removing or exchanging any rows or columns cause changes of the above numeration.

B. when it is a negative number or equal to 0 it controls the printout by calling one of implemented module functions.

Field 2

The contents of description field destined for the printout being :

- in case A - a description of data item,
- in case B - a heading.

Field 3

Dimension field contents is a text describing dimension of output data. If the data have to be non-dimensional the field should contain left aligned "N.D." string.

Field 4

The field contains a factor by which data are to be multiplied before being located in the printout.

Field 5

The field contains a format description for REAL numbers according to FORTRAN standard. The format refers to one value and a field 16 character long . Output data will be rounded according to the given format.

Field 6

If the field contains "1" a row will be printed out even if the rounded data value is equal to 0. If it contains "2" such a row is omitted in the printout.

Field 7

The field contains MPS code of the row being a reference for certain module functions what will be discussed further on.

Module actions in case of omitting any of the fields (the field is filled by space characters) are as follows:

Field 1.

Action as for an indicator equal to 0.

Field 2.

In appropriate place 40 space characters will be printed.

Field 3-6.

Default contents of field fixed earlier will be accepted and an appropriate action will be performed. The default contents are fixed at the beginning of the run; they may be also changed by filling appropriate fields in DICTIONARY file records with an indicator less or equal to 0.

Default contents of fields are:

Field	Default	Description
3	N.D.	output value is non-dimensional
4	1.	scale of output value remains unchanged
5	(F16.0)	adequately rounded integer parts of number will be put out
6	2	value equal to 0 after rounding will be neglected
7		row indicated is presently an objective function

Therefore when a printout row is being completed (case A) and any of the fields is omitted, the contents fixed by the last control record will be accepted (case B) or if this field was omitted in all control records recognized so far - the fixed primary value will be accepted.

The module ends its work after having met the end of file marker of DICTIONARY file.

3.2.5. Implemented module functions.

In its current version the module performs the following functions denoted below by their coding indicator (field 1):

0 printout of field 2 from a new page preceded and followed by 3 blank lines;

1 like in 0 function but without shift to new line;

11 printout of primary solution with single shift;

12 printout of primary solution with percentage, that is, the ratio of variable and its upper constraint, with a single shift;

13 like in "11" function with MPS codes printout for the variables if they exist;

14 like in "12" function with MPS codes printout for the variables if they exist;

15 printout of primary solution for structural variables with single shift and with the following information:

- product of variable and coefficient of the objective function with scale and rounding according to contents of fields 4 and 5.
- quotient of coefficient by a variable in the objective function and coefficient in the row indicated by field 7.

If for arithmetical reasons it is impossible to obtain any of printout values (division by 0) or searched coefficient appears to be equal to 0 - the module changes the printout format (omits appropriate fields) automatically.

Other functions apart from the described above may be implemented only after making necessary changes and extensions.

3.2.6. Emergency situations in module processing.

The module run will be interrupted and appropriate information given in case the optimal solution has not been found.

All other errors refer to the situation that occurred during analysis of consecutive records of DICTIONARY file. The module ignores a record that appeared to be incorrect writing out an appropriate information and passes on to analyzing the next record. The number of errors of this type is counted. In case that number surpasses the declared error quantity (declaration as for MPS file) - the module will stop its action.

3.2.7. An Example

As an example, results of the same LFP programming problem as previously (see point 2.4) are processed by the module. An exemplary report with an adequate DICTIONARY file is enclosed below.

```
      -1EXAMPLARY RESULTS
      -1Values of Rows
ob      Numerator Value (ob)                UnitOfob
de      Denominator Value (de)              UnitOfde
r1      First Row (r1)                      UnitOfr1
r2      Second Row (r2)                     n.d.
r3      Third Row (r3)                      UnitOfr3
r4      Fourth Row (r4)                     UnitOfr4
      -1Values of Columns                                  (f16.2)
x3      Third Column (x3)                   UnitOfx3
x2      Second Column (x2)                  UnitOfx2
x1      First Column (x1)                   UnitOfx1


          m i n o s    ---    version  4.0    mar  1981
          = = = = =

specs file
----------
         begin POSTAN - Example of Report Option
              maximize
              objective ob
              denominator de
         *
         *    Previously obtained solution will be printed by the option.
         *
              report yes
              dictionary file 11
         end

**********************************************************************************

problem  test      objective  ob      /de       (max)  value     2.8865671064d+00

**********************************************************************************




        EXEMPLARY RESULTS




        Values of Rows



Numerator Value (ob)                          29. UnitOfob
```

```
Denominator Value (de)                               10. UnitOfde
Second Row (r2)                                      -7.
Third Row (r3)                                        4. UnitOfr3


         Values of Columns


Second Column (x2)                                -1.40 UnitOfx2
First Column (x1)                                  1.41 UnitOfx1



********************************************************************************
```

4. REFERENCES

1. G.Dobrowolski, K.Hajduk, A.Korytowski and T.Rys: *POSTAN - A Package for Postoptimal Analysis (An Extension of MINOS)*. International Institute for Applied Systems Analysis, IIASA Collaborative Paper CP-84-32, Laxenburg, July 1984.
2. G.Dobrowolski, K.Hajduk, A.Korytowski and T.Rys: *POSTAN 2 - An Extended Postoptimal Analysis Package for MINOS* , In Theory, Software and Testing Examples for Decision Support Systems, A. Wierzbicki and A. Lewandowski Eds., International Institute for Applied Systems Analysis, Laxenburg, 1985.
3. B.A.Murtagh and M.A.Saunders: *MINOS - A Large-Scale Linear Programming System. User s Guide*. Technical Report SOL 77-9, System Optimization Laboratory, Stanford University, California 1977.