H02BUF – NAG Fortran Library Routine Document

Note. Before using this routine, please read the Users' Note for your implementation to check the interpretation of bold italicised terms and other implementation-dependent details.

1 Purpose

H02BUF reads data for a linear or integer programming problem from an external file which is in standard or compatible MPSX input format.

2 Specification

```
      SUBROUTINE H02BUF(INFILE, MAXN, MAXM, OPTIM, XBLDEF, XBUDEF,
     1                  NMOBJ, NMRHS, NMRNG, NMBND, MPSLST, N, M, A, BL,
     2                  BU, CVEC, X, INTVAR, CRNAME, NMPROB, IWORK, IFAIL)
      INTEGER           INFILE, MAXN, MAXM, N, M, INTVAR(MAXN),
     1                  IWORK(MAXN+MAXM), IFAIL
      real              XBLDEF, XBUDEF, A(MAXM,MAXN), BL(MAXN+MAXM),
     1                  BU(MAXN+MAXM), CVEC(MAXN), X(MAXN)
      CHARACTER*3       OPTIM
      CHARACTER*8       NMOBJ, NMRHS, NMRNG, NMBND, CRNAME(MAXN+MAXM),
     1                  NMPROB
      LOGICAL           MPSLST
```

3 Description

H02BUF reads linear programming (LP) or integer programming (IP) problem data from an external file which is prepared in standard or compatible MPSX [1] input format and then initializes n (the number of variables), m (the number of general linear constraints), the vectors c, l and u and the m by n matrix A for use with E04MFF or H02BBF, which are designed to solve problems of the form

$$\underset{x \in R^n}{\text{minimize}}\ c^T x \ \text{ subject to } \ l \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u.$$

This routine may be followed by calls to either E04MFF (to solve an LP problem) or H02BBF and H02BZF (to solve an IP problem), possibly followed by a call to H02BVF (to print the solution using MPSX names).

Note that H02BUF uses an 'infinite' bound size of 10^{20} in the definition of l and u. In other words, any element of u greater than or equal to 10^{20} will be regarded as $+\infty$ (and similarly any element of l less than or equal to -10^{20} will be regarded as $-\infty$). If this value is deemed to be 'inappropriate', users are recommended to reset the value of either the optional parameter **Infinite BoundSize** (if an LP problem is being solved) or the parameter BIGBND (if an IP problem is being solved) and make any necessary changes to BL and/or BU prior to calling E04MFF or H02BBF (as appropriate).

The documents for H02BVF, E04MFF and/or H02BBF and H02BZF should be consulted for further details.

MPSX input format

The input file of data may only contain two types of lines.

(1) Indicator lines (specifying the type of data which is to follow).
(2) Data lines (specifying the actual data).

The input file must not contain any blank lines. Any characters beyond column 80 are ignored. Indicator lines must not contain leading blank characters (in other words they must begin in column 1). The following displays the order in which the indicator lines must appear in the file:

NAME user-given name
ROWS
  data line(s)
COLUMNS
  data line(s)
RHS
  data line(s)
RANGES (optional)
  data line(s)
BOUNDS (optional)
  data line(s)
ENDATA

The 'user-given name' specifies a name for the problem and must occupy columns 15–22. The name can either be blank or up to a maximum of 8 characters.

A data line follows the same fixed format made up of fields defined below. The contents of the fields may have different significance depending upon the section of data in which they appear.

	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
Columns	2–3	5–12	15–22	25–36	40–47	50–61
Contents	Code	Name	Name	Value	Name	Value

The names and codes consist of 'alphanumeric' characters (i.e., a–z, A–Z, 0–9, +, −, asterisk (*), blank ( ), colon (:), dollar sign ($) or fullstop (.) only) and the names must not contain leading blank characters. Values are read using Fortran format E12.0. This allows values to be entered in several equivalent forms. For example, 1.2345678, 1.2345678E+0, 123.45678E−2 and 12345678E−07 all represent the same number. It is safest to include an explicit decimal point.

Note that in order to ensure numeric values are interpreted as intended, they should be *right-justified* in the 12-character field, with no trailing blanks. This is because in some situations trailing blanks may be interpreted as zeros and this can dramatically affect the interpretation of the value. This is relevant if the value contains an exponent, or if it contains neither an exponent nor an explicit decimal point. For example, the fields

```
%%%%1.23E-2%
%%%%%%%123%%
```

may be interpreted as 1.23E−20 and 12300 respectively (where % is used to denote a blank). The actual behaviour is system-dependent.

Comment lines are allowed in the data file. These must have an asterisk (*) in column 1 and any characters in columns 2–80. In any data line, a dollar sign ($) as the first character in field 3 or 5 indicates that the information from that point through column 80 consists of comments.

Columns outside the six fields must be blank, except for columns 72–80, whose contents are ignored by the routine. These columns may be used to enter a sequence number. A non-blank character outside the predefined six fields and columns 72–80 is considered to be a major error (IFAIL = 11; see Section 6), unless it is part of a comment.

ROWS Data Lines

These lines specify row (constraint) names and their inequality types (i.e., =, $\geq$ or $\leq$).

Field 1:	defines the constraint type. It may be in column 2 or column 3.
N	free row, that is no constraint. It may be used to define the objective row.
G	greater than or equal to (i.e., $\geq$).
L	less than or equal to (i.e., $\leq$).
E	exactly equal to (i.e., =).
Field 2:	defines the row name.

Row type N stands for 'Not binding', also known as 'Free'. It can be used to define the objective row. The objective row is a free row that specifies the vector c in the objective function. It is taken to be the first free row, unless some other free row name is specified by the parameter NMOBJ (see Section 5). Note that the objective function must be included in the MPSX data file. Thus the maximum number of constraints (MAXM; see Section 5) in the problem must be $m + 1$.

COLUMNS Data Lines

These lines specify the names to be assigned to the variables (columns) in the constraint matrix A, and define, in terms of column vectors, the actual values of the corresponding matrix elements.

Field 1: blank (ignored)

Field 2: gives the name of the column associated with the elements specified in the following fields.

Field 3: contains the name of a row.

Field 4: used in conjunction with field 3 contains the value of the matrix element.

Field 5: is optional (may be used like field 3).

Field 6: is optional (may be used like field 4).

Note that only non-zero elements of A need to be specified in the COLUMNS section, as any unspecified elements are assumed to be zero.

RHS Data Lines

This section specifies the right-hand side values of the constraint matrix A. The lines specify the name of the RHS (right-hand side) vector given to the problem, the numerical values of the elements of the vector are also defined by the data lines and may appear in any order. The data lines have exactly the same format as the COLUMNS data lines, except that the column name is replaced by the RHS name. Note that any unspecified elements are assumed to be zero.

RANGES Data Lines (optional)

Ranges are used for constraints of the form $l \leq Ax \leq u$, where l and u are finite. The range of the constraint is $r = u - l$. Either l or u must be specified in the RHS section and r must be defined in this section.

The data lines have exactly the same format as the COLUMNS data lines, except that the column name is replaced by the RANGE name.

BOUNDS Data Lines (optional)

These lines specify limits on the values of the variables (l and u in $l \leq x \leq u$). If the variable is not specified in the bound set then it is automatically assumed to lie between default lower and upper bounds (usually 0 and $+\infty$).
Like an RHS column which is given a name, the set of variables in one bound set is also given a name.

Field 1:	specifies the type of bound or defines the variable type.
`LO`	lower bound
`UP`	upper bound
`FX`	fixed variable
`FR`	free variable ($-\infty$ to $+\infty$)
`MI`	lower bound is $-\infty$
`PL`	upper bound is $+\infty$. This is the default variable type.
Field 2:	identifies a name for the bound set.
Field 3:	identifies the column name of the variable belonging to this set.
Field 4:	identifies the value of the bound; this has a numerical value only in association with `LO`, `UP`, `FX` in field 1, otherwise it is blank.
Field 5:	is blank and ignored.
Field 6:	is blank and ignored.

Note that if RANGES and BOUNDS sections are both present, the RANGES section must appear first.

Integer problems

In IP problems there are two common integer variable types. (a) 0–1 integer variables which represent 'on' or 'off' situations and (b) General integer variables which are forced to take an integer value, in a specified range, at the optimal integer solution. Integer variables can be defined in the following compatible and standard MPSX forms.

In the compatible MPSX format, the type of integer variables are defined in Field 1 of the BOUNDS section, that is:

Field 1:	specifies the type of the integer variable.
`BV`	0–1 integer variable (bound value is 1.0).
`UI`	general integer variable (bound value is in Field 4).

In the standard MPSX format, the integer variables are treated the same as the 'ordinary' bounded variables, in the BOUNDS section. Integer markers are, however, introduced in the COLUMNS section to specify the integer variables. The indicator lines for these markers are:

	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
Columns	2–3	5–12	15–22	25–36	40–47	50–61
Contents		INTEGER	'MARKER'		'INTORG'	

to mark the beginning of the integer variables and

	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
Columns	2–3	5–12	15–22	25–36	40–47	50–61
Contents		INTEGER	'MARKER'		'INTEND'	

to mark the end. That is, any variables between these markers are treated as integer variables. Note that if the (INTEND) indicator line is not specified in the file then all the variables between the (INTORG) indicator line and the end of the COLUMNS section are assumed to be integer variables. The routine accepts both standard and/or compatible MPSX format as a means of specifying integer variables. This is illustrated in Section 9.2 of the document for H02BFF.

4 References

[1] (1971) MPSX – Mathematical programming system *Program Number 5734 XM4* IBM Trade Corporation, New York

5 Parameters

1: INFILE — INTEGER *Input*

On entry: the unit number associated with the MPSX data file.

Constraint: $0 \leq$ INFILE ≤ 99.

2: MAXN — INTEGER *Input*

On entry: an upper limit for the number of variables in the problem.

Constraint: MAXN ≥ 1.

3: MAXM — INTEGER *Input*

On entry: an upper limit for the number of constraints (including the objective) in the problem.

Constraint: MAXM ≥ 1.

4: OPTIM — CHARACTER*3 *Input*

On entry: specifies the direction of the optimization. OPTIM must be set to 'MIN' for minimization and to 'MAX' for maximization.

Constraint: OPTIM = 'MIN' or 'MAX'.

5: XBLDEF — ***real*** *Input*

On entry: the default lower bound to be used for the variables in the problem when none is specified in the BOUNDS section of the MPSX data file. For a standard LP or IP problem XBLDEF would normally be set to zero.

6: XBUDEF — ***real*** *Input*

On entry: the default upper bound to be used for the variables in the problem when none is specified in the BOUNDS section of the MPSX data file. For a standard LP or IP problem XBUDEF would normally be set to 'infinity' (i.e., XBUDEF $\geq 10^{20}$).

Constraint: XBUDEF $\geq$ XBLDEF.

7: NMOBJ — CHARACTER*8 *Input/Output*

On entry: either the name of the objective function to be used for the optimization, or blank (in which case the first objective (free) row in the file is used).

On exit: the name of the objective row as defined in the MPSX data file.

8: NMRHS — CHARACTER*8 *Input/Output*

On entry: either the name of the RHS set to be used for the optimization, or blank (in which case the first RHS set is used).

On exit: the name of the RHS set read in the MPSX data file.

9: NMRNG — CHARACTER*8 *Input/Output*

On entry: either the name of the RANGE set to be used for the optimization, or blank (in which case the first RANGE set (if any) is used).

On exit: the name of the RANGE set read in the MPSX data file. This is blank if the MPSX data file does not have a RANGE set.

10: NMBND — CHARACTER*8 *Input/Output*

On entry: either the name of the BOUNDS set to be used for the optimization, or blank (in which case the first BOUNDS set (if any) is used).

On exit: the name of the BOUNDS set read in the MPSX data file. This is blank if the MPSX data file does not have a BOUNDS set.

11: MPSLST — LOGICAL *Input*

On entry: if MPSLST = .TRUE., then a listing of the input data is sent to the current advisory message unit (as defined by X04ABF). This can be useful for debugging the MPSX data file.

12: N — INTEGER *Output*

On exit: n, the actual number of variables in the problem.

13: M — INTEGER *Output*

On exit: m, the actual number of general linear constraints in the problem.

14: A(MAXM,MAXN) — ***real*** array *Output*

On exit: A, the matrix of general linear constraints.

15: BL(MAXN+MAXM) — ***real*** array *Output*

On exit: l, the lower bounds for all the variables and constraints in the following order. The first N elements of BL contain the bounds on the variables and the next M elements contain the bounds for the general linear constraints (if any). Note that an 'infinite' lower bound is indicated by BL(j) = −1.0E+20 and an equality constraint by BL(j) = BU(j).

16: BU(MAXN+MAXM) — ***real*** array *Output*

On exit: u, the upper bounds for all the variables and constraints in the following order. The first N elements of BU contain the bounds on the variables and the next M elements contain the bounds for the general linear constraints (if any). Note that an 'infinite' upper bound is indicated by BU(j) = 1.0E+20 and an equality constraint by BU(j) = BL(j).

17: CVEC(MAXN) — ***real*** array *Output*

On exit: c, the coefficients of the objective function. The signs of these coefficients are determined by the problem (either LP or IP) and the direction of the optimization (see OPTIM above).

18: X(MAXN) — ***real*** array *Output*

On exit: an initial estimate of the solution to the problem. More precisely, X(j) = 1.0 if j is odd and 0.0 otherwise, for $j = 1, 2, \ldots$,N.

19: INTVAR(MAXN) — INTEGER array *Output*

On exit: indicates which are the integer variables in the problem. More precisely, INTVAR(k) = 1 if x_k is an integer variable, and 0 otherwise, for k = 1,2,...,N.

20: CRNAME(MAXN+MAXM) — CHARACTER*8 array *Output*

On exit: the MPSX names of all the variables and constraints in the problem in the following order. The first N elements contain the MPSX names for the variables and the next M elements contain the MPSX names for the general linear constraints (if any).

21: NMPROB — CHARACTER*8 *Output*

On exit: the name of the problem as defined in the MPSX data file.

22: IWORK(MAXN+MAXM) — INTEGER array *Workspace*

23: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, −1 or 1. For users not familiar with this parameter (described in Chapter P01) the recommended value is 0.

On exit: IFAIL = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

If on entry IFAIL = 0 or −1, explanatory error messages are output on the current error message unit (as defined by X04AAF).

Errors detected by the routine:

IFAIL = 1

There are too many rows present in the data file. Increase MAXM by at least (M − MAXM) and rerun H02BUF.

IFAIL = 2

There are too many columns present in the data file. Increase MAXN by at least (N − MAXN) and rerun H02BUF.

The following error exits (apart from IFAIL = 14) are caused by having either a corrupt or a non-standard MPSX data file. Refer to Section 3 for a detailed description of the MPSX format which can be read by H02BUF. If MPSLST = .TRUE., the last line of printed output refers to the line in the MPSX data file which contains the reported error.

IFAIL = 3

The objective function row was not found. There must be at least one row in the ROWS section with row type N for the objective row.

IFAIL = 4

There are no rows specified in the ROWS section.

IFAIL = 5

An illegal constraint type was detected in the ROWS section. The constraint type must be one of N, L, G or E.

IFAIL = 6

An illegal row name was detected in the ROWS section. Names must be made up of alphanumeric characters with no leading blanks.

IFAIL = 7

An illegal column name was detected in the COLUMNS section. Names must be made up of alphanumeric characters with no leading blanks.

IFAIL = 8

An illegal bound type was detected in the BOUNDS section. The bound type must be one of LO, UP, FX, FR, MI, PL, BV or UI.

IFAIL = 9

An unknown column name was detected in the BOUNDS section. All the column names must be specified in the COLUMNS section.

IFAIL = 10

The last line in the file does not contain the ENDATA line indicator.

IFAIL = 11

An illegal data line was detected in the file. This line is neither a comment line nor a valid data line.

IFAIL = 12

An unknown row name was detected in COLUMNS or RHS or RANGES section. All the row names must be specified in the ROWS section.

IFAIL = 13

There were no columns specified in the COLUMNS section.

IFAIL = 14

An input parameter is invalid.

IFAIL = 15

Incorrect integer marker. In standard MPSX data format, integer variables should be defined between INTORG and INTEND markers.

7 Accuracy

Not applicable.

8 Further Comments

None.

9 Example

This example solves the same problem as the example for H02BFF, except that it treats it as an LP problem.

One of the applications of linear programming is to the so-called diet problem. Given the nutritional content of a selection of foods, the cost of each food, the amount available of each food and the consumer's minimum daily energy requirements, the problem is to find the cheapest combination. This gives rise to the following problem:

minimize

$$c^T x$$

subject to

$$Ax \geq b,$$
$$0 \leq x \leq u,$$

where

$$c = (3\ 24\ 13\ 9\ 20\ 19)^T,\ x = (x_1, x_2, x_3, x_4, x_5, x_6)^T \text{ is real,}$$

$$A = \begin{pmatrix} 110 & 205 & 160 & 160 & 420 & 260 \\ 4 & 32 & 13 & 8 & 4 & 14 \\ 2 & 12 & 54 & 285 & 22 & 80 \end{pmatrix}, \quad b = \begin{pmatrix} 2000 \\ 55 \\ 800 \end{pmatrix}$$

and $u = (4\ 3\ 2\ 8\ 2\ 2)^T$.

The rows of A correspond to energy, protein and calcium and the columns of A correspond to oatmeal, chicken, eggs, milk, pie and bacon respectively.

The MPSX representation of the problem is given in Section 9.2.

9.1 Program Text

Note. The listing of the example program presented below uses bold italicised terms to denote precision-dependent details. Please read the Users' Note for your implementation to check the interpretation of these terms. As explained in the Essential Introduction to this manual, the results produced may not be identical for all implementations.

```
*     H02BUF Example Program Text
*     Mark 16 Release. NAG Copyright 1993.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          MAXN, MAXM
      PARAMETER        (MAXN=50,MAXM=50)
      INTEGER          LDA
      PARAMETER        (LDA=MAXM)
      real             XBUDEF, XBLDEF
      PARAMETER        (XBUDEF=1.0e+20,XBLDEF=0.0e0)
      INTEGER          LIWORK
      PARAMETER        (LIWORK=2*MAXN+3)
      INTEGER          LWORK
      PARAMETER        (LWORK=2*(MAXM+1)**2+7*MAXN+5*MAXM)
      CHARACTER*3      OPTIM
      PARAMETER        (OPTIM='MIN')
*     .. Local Scalars ..
      real             OBJVAL
      INTEGER          IFAIL, INFILE, ITER, M, N
      LOGICAL          MPSLST
      CHARACTER*8      KBLANK, NMBND, NMOBJ, NMPROB, NMRHS, NMRNG
*     .. Local Arrays ..
      real             A(MAXM,MAXN), AX(MAXM), BL(MAXN+MAXM),
     +                 BU(MAXN+MAXM), CLAMDA(MAXN+MAXM), CVEC(MAXN),
     +                 WORK(LWORK), X(MAXN)
```

```
      INTEGER          INTVAR(MAXN), ISTATE(MAXN+MAXM), IWORK(LIWORK)
      CHARACTER*8      CRNAME(MAXN+MAXM)
*     .. External Subroutines ..
      EXTERNAL         E04MFF, E04MHF, H02BUF, H02BVF
*     .. Data statements ..
      DATA             KBLANK/'        '/
*     .. Executable Statements ..
      WRITE (NOUT,*) 'H02BUF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
*
*     Initialize parameters
*
      INFILE = NIN
      NMPROB = KBLANK
      NMOBJ = KBLANK
      NMRHS = KBLANK
      NMRNG = KBLANK
      NMBND = KBLANK
      MPSLST = .FALSE.
*
      IFAIL = 0
*
*     Convert the MPSX data file for use by E04MFF
*
      CALL H02BUF(INFILE,MAXN,MAXM,OPTIM,XBLDEF,XBUDEF,NMOBJ,NMRHS,
     +            NMRNG,NMBND,MPSLST,N,M,A,BL,BU,CVEC,X,INTVAR,CRNAME,
     +            NMPROB,ISTATE,IFAIL)
*
*     Solve the problem
*
      IFAIL = -1
*
      CALL E04MHF('Print Level = 5')
*
      CALL E04MFF(N,M,A,LDA,BL,BU,CVEC,ISTATE,X,ITER,OBJVAL,AX,CLAMDA,
     +            IWORK,LIWORK,WORK,LWORK,IFAIL)
*
      IF (IFAIL.EQ.0 .OR. IFAIL.EQ.1 .OR. IFAIL.EQ.3) THEN
*
*        Print solution (using MPSX names)
*
         IFAIL = 0
*
         CALL H02BVF(N,M,A,LDA,BL,BU,X,CLAMDA,ISTATE,CRNAME,IFAIL)
*
      ELSE
         WRITE (NOUT,99999) 'E04MFF terminated with IFAIL = ', IFAIL
      END IF
*
      STOP
*
99999 FORMAT (1X,A,I3)
      END
```

9.2 Program Data

```
H02BUF Example Program Data
NAME          DIET
ROWS
 G  ENERGY
 G  PROTEIN
 G  CALCIUM
 N  COST
COLUMNS
    OATMEAL   ENERGY      110.0
    OATMEAL   PROTEIN     4.0
    OATMEAL   CALCIUM     2.0
    OATMEAL   COST        3.0
    CHICKEN   ENERGY      205.0
    CHICKEN   PROTEIN     32.0
    CHICKEN   CALCIUM     12.0
    CHICKEN   COST        24.0
    EGGS      ENERGY      160.0
    EGGS      PROTEIN     13.0
    EGGS      CALCIUM     54.0
    EGGS      COST        13.0
    MILK      ENERGY      160.0
    MILK      PROTEIN     8.0
    MILK      CALCIUM     285.0
    MILK      COST        9.0
    PIE       ENERGY      420.0
    PIE       PROTEIN     4.0
    PIE       CALCIUM     22.0
    PIE       COST        20.0
    BACON     ENERGY      260.0
    BACON     PROTEIN     14.0
    BACON     CALCIUM     80.0
    BACON     COST        19.0
RHS
    DEMANDS   ENERGY      2000.0
    DEMANDS   PROTEIN     55.0
    DEMANDS   CALCIUM     800.0
BOUNDS
 UI SERVINGS  OATMEAL     4.0
 UI SERVINGS  CHICKEN     3.0
 UP SERVINGS  EGGS        2.0
 UP SERVINGS  MILK        8.0
 UP SERVINGS  PIE         2.0
 UI SERVINGS  BACON       2.0
ENDATA
```

9.3 Program Results

```
H02BUF Example Program Results


Calls to E04MHF
---------------

     Print Level = 5

*** E04MFF
*** Start of NAG Library implementation details ***

Implementation title: Generalised Base Version
           Precision: FORTRAN double precision
        Product Code: FLBAS19D
                Mark: 19A

*** End of NAG Library implementation details ***

Parameters
----------

Problem type...........        LP

Linear constraints.....         3       Feasibility tolerance..  1.05E-08
Variables..............         6       Optimality tolerance...  1.72E-13

Infinite bound size....  1.00E+20       COLD start.............
Infinite step size.....  1.00E+20       EPS (machine precision)  1.11E-16

Check frequency........        50       Expand frequency.......         5
Minimum sum of infeas..        NO       Crash tolerance........  1.00E-02

Print level............         5       Iteration limit........        50
Monitoring file........        -1

Workspace provided is     IWORK(     103),  WORK(    5802).
To solve problem we need  IWORK(      15),  WORK(      89).


 Itn     Step Ninf Sinf/Objective  Norm Gz
   0  0.0E+00    3   1.799000E+03  0.0E+00
   1  1.5E-02    1   2.550000E+02  0.0E+00
   2  1.4E-03    0   1.271429E+02  0.0E+00
   3  8.7E-02    0   1.129048E+02  0.0E+00
   4  2.1E-01    0   1.062857E+02  0.0E+00
   5  1.9E+00    0   9.733333E+01  0.0E+00
   6  2.9E+00    0   9.250000E+01  0.0E+00

Exit E04MFF - Optimal LP solution.

Final LP objective value =    92.50000

Exit from LP problem after     6 iterations.


Varbl   State     Value     Lower Bound   Upper Bound    Lagr Mult   Residual
```

```
OATMEAL  UL    4.00000      0.000000E+00  4.00000      -3.187      0.0000E+00
CHICKEN  LL   0.000000E+00  0.000000E+00  3.00000       12.47      0.0000E+00
EGGS     LL   0.000000E+00  0.000000E+00  2.00000       4.000      0.0000E+00
MILK     FR    4.50000      0.000000E+00  8.00000      0.0000E+00   3.500
PIE      UL    2.00000      0.000000E+00  2.00000      -3.625      0.0000E+00
BACON    LL   0.000000E+00  0.000000E+00  2.00000       4.375      0.0000E+00


L Con   State     Value     Lower Bound  Upper Bound   Lagr Mult   Residual

ENERGY   LL    2000.00       2000.00         None      5.6250E-02  0.0000E+00
PROTEIN  FR    60.0000       55.0000         None      0.0000E+00   5.000
CALCIUM  FR    1334.50       800.000         None      0.0000E+00   534.5
```