H02BBF – 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.

Warning. The specification of the parameters BIGBND and LIWORK changed at Mark 16: the 'default' value of the parameter BIGBND has been increased to 10^{20} and the minimum dimension of the array IWORK has been increased by N + 3.

1 Purpose

H02BBF solves 'zero-one', 'general', 'mixed' or 'all' integer programming problems using a branch and bound method. The routine may also be used to find either the first integer solution or the optimum integer solution. It is not intended for large sparse problems.

2 Specification

```
 SUBROUTINE H02BBF(ITMAX, MSGLVL, N, M, A, LDA, BL, BU, INTVAR,
1                  CVEC, MAXNOD, INTFST, MAXDPT, TOLIV, TOLFES,
2                  BIGBND, X, OBJMIP, IWORK, LIWORK, RWORK, LRWORK,
3                  IFAIL)
 INTEGER           ITMAX, MSGLVL, N, M, LDA, INTVAR(N), MAXNOD,
1                  INTFST, MAXDPT, IWORK(LIWORK), LIWORK, LRWORK,
2                  IFAIL
 real              A(LDA,*), BL(N+M), BU(N+M), CVEC(N), TOLIV,
1                  TOLFES, BIGBND, X(N), OBJMIP, RWORK(LRWORK)
```

3 Description

H02BBF is capable of solving certain types of integer programming (IP) problems using a branch and bound (B&B) method, see Taha [1]. In order to describe these types of integer programs and to briefly state the B&B method, we define the following linear programming (LP) problem:

Minimize

$$F(x) = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

subject to

$$\sum_{j=1}^{n} a_{ij}x_j \left\{ \begin{array}{c} = \\ \leq \\ \geq \end{array} \right\} b_i, \quad i = 1, 2, \ldots, m$$

$$l_j \leq x_j \leq u_j, \qquad j = 1, 2, ..., n \tag{1}$$

If, in (1), it is required that (some or) all the variables take integer values, then the integer program is of type (*mixed* or) *all* general IP problem. If additionally, the integer variables are restricted to take only 0-1 values (i.e., $l_j = 0$ and $u_j = 1$) then the integer program is of type (mixed or all) *zero-one* IP problem.

The B&B method applies directly to these integer programs. The general idea of B&B (for a full description see Dakin [2] or Mitra [3]) is to solve the problem without the integral restrictions as an LP problem (first *node*). If in the optimal solution an integer variable x_k takes a non-integer value x_k^*, two LP sub-problems are created by *branching*, imposing $x_k \leq [x_k^*]$ and $x_k \geq [x_k^*] + 1$ respectively, where $[x_k^*]$ denotes the integer part of x_k^*. This method of branching continues until the first integer solution (*bound*) is obtained. The hanging nodes are then solved and investigated in order to prove the optimality of the solution. At each node, an LP problem is solved using E04MFF.

4 References

[1] Taha H A (1987) *Operations Research: An Introduction* Macmillan, New York

[2] Dakin R J (1965) A tree search algorithm for mixed integer programming problems *Comput. J.* **8** 250–255

[3] Mitra G (1973) Investigation of some branch and bound strategies for the solution of mixed integer linear programs *Math. Programming* **4** 155–170

5 Parameters

1: ITMAX — INTEGER *Input/Output*

On entry: an upper bound on the number of iterations for each LP problem.

On exit: unchanged if on entry ITMAX > 0, else ITMAX = $\max(50, 5\times$ (N + M)).

2: MSGLVL — INTEGER *Input*

On entry: the amount of printout produced by H02BBF, as indicated below (see Section 5.1 for a description of the printed output). All output is written to the current advisory message unit (as defined by X04ABF).

Value	**Definition**
0	No output.
1	The final IP solution only.
5	One line of output for each node investigated and the final IP solution.
10	The original LP solution (first node), one line of output for each node investigated and the final IP solution.

3: N — INTEGER *Input*

On entry: n, the number of variables.

Constraint: N > 0.

4: M — INTEGER *Input*

On entry: m, the number of general linear constraints.

Constraint: M $\geq$ 0.

5: A(LDA,$*$) — ***real*** array *Input*

Note: the second dimension of the array A must be at least N when M > 0, and at least 1 when M = 0.

On entry: the ith row of A must contain the coefficients of the ith general constraint, for $i = 1, 2, \ldots, m$.

If M = 0 then the array A is not referenced.

6: LDA — INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which H02BBF is called.

Constraint: LDA $\geq$ max(1,M).

7: BL(N+M) — ***real*** array *Input*

8: BU(N+M) — ***real*** array *Input*

On entry: BL must contain the lower bounds and BU the upper bounds, for all the constraints in the following order. The first n elements of each array must contain the bounds on the variables, and the next m elements the bounds for the general linear constraints (if any). To specify a non-existent lower bound (i.e., $l_j = -\infty$), set BL(j) $\leq$ $-$BIGBND and to specify a non-existent upper bound (i.e., $u_j = +\infty$), set BU(j) $\geq$ BIGBND. To specify the jth constraint as an equality, set BL(j) = BU(j) = β, say, where $|\beta| <$ BIGBND.

Constraints:

BL(j) $\leq$ BU(j), for $j = 1, 2, \ldots$,N + M,
$|\beta| <$ BIGBND when BL(j) = BU(j) = β.

9: INTVAR(N) — INTEGER array *Input*

On entry: indicates which are the integer variables in the problem. For example, if x_j is an integer variable then INTVAR(j) must be set to 1, and 0 otherwise.

Constraints:

INTVAR}(j) = 0 or 1 for $j = 1, 2, \ldots$,N, and
INTVAR(j) = 1 for at least one value of j.

10: CVEC(N) — ***real*** array *Input*

On entry: the coefficients c_j of the objective function $F(x) = c_1x_1 + c_2x_2 + \ldots + c_nx_n$. The routine attempts to find a minimum of $F(x)$. If a maximum of $F(x)$ is desired, CVEC(j) should be set to $-c_j$, for $j = 1, 2, \ldots, n$, so that the routine will find a minimum of $-F(x)$.

11: MAXNOD — INTEGER *Input*

On entry: the maximum number of nodes that are to be searched in order to find a solution (optimum integer solution). If MAXNOD ≤ 0 and INTFST ≤ 0, then the B&B tree search is continued until all the nodes have been investigated.

12: INTFST — INTEGER *Input*

On entry: specifies whether to terminate the B&B tree search after the first integer solution (if any) is obtained. If INTFST > 0 then the B&B tree search is terminated at node k say, which contains the first integer solution. For MAXNOD > 0 this applies only if $k \leq$ MAXNOD.

13: MAXDPT — INTEGER *Input*

On entry: the maximum depth of the B&B tree used for branch and bound.

Suggested value: MAXDPT = 3 $\times$ N/2.

Constraint: MAXDPT ≥ 2.

14: TOLIV — ***real*** *Input/Output*

On entry: the integer feasibility tolerance; i.e., an integer variable is considered to take an integer value if its violation does not exceed TOLIV. For example, if the integer variable x_j is of order unity then x_j is considered to be integer only if (1-TOLIV) $\leq x_j \leq$ (1+TOLIV).

On exit: unchanged if on entry TOLIV > 0.0, else TOLIV = 10^{-5}.

15: TOLFES — ***real*** *Input/Output*

On entry: the maximum acceptable absolute violation in each constraint at a 'feasible' point (feasibility tolerance); i.e., a constraint is considered satisfied if its violation does not exceed TOLFES.

On exit: unchanged if on entry TOLFES > 0.0, else TOLFES = $\sqrt{\epsilon}$ (where ϵ is the ***machine precision***).

16: BIGBND — ***real*** *Input/Output*

On entry: the 'infinite' bound size in the definition of the problem constraints. More precisely, any upper bound greater than or equal to BIGBND will be regarded as $+\infty$ and any lower bound less than or equal to $-$BIGBND will be regarded as $-\infty$.

On exit: unchanged if on entry BIGBND > 0.0, else BIGBND $= 10^{20}$.

17: X(N) — ***real*** array *Input/Output*

On entry: an initial estimate of the original LP solution.

On exit: with IFAIL = 0, X contains a solution which will be an estimate of either the optimum integer solution or the first integer solution, depending on the value of INTFST. If IFAIL = 9, then X contains a solution which will be an estimate of the best integer solution that was obtained by searching MAXNOD nodes.

18: OBJMIP — ***real*** *Output*

On exit: with IFAIL = 0 or 9, OBJMIP contains the value of the objective function for the IP solution.

19: IWORK(LIWORK) — INTEGER array *Workspace*

20: LIWORK — INTEGER *Input*

On entry: the dimension of the array IWORK as declared in the (sub)program from which H02BBF is called.

Constraint: LIWORK $\geq$ (25+N+M) $\times$ MAXDPT + 5 s N + M + 4.

21: RWORK(LRWORK) — ***real*** array *Workspace*

22: LRWORK — INTEGER *Input*

On entry: the dimension of the array RWORK as declared in the (sub)program from which H02BBF is called.

Constraint: LRWORK $\geq$ MAXDPT $\times$ (N+2) + 2 $\times$ {MIN}(N,M+1)^2} + 13 $\times$ N + 12 $\times$ M.

If MSGLVL > 0, the amounts of workspace provided and required (with MAXDPT = 3 $\times$ N/2) are printed. As an alternative to computing MAXDPT, LIWORK and LRWORK from the formulas given above, the user may prefer to obtain appropriate values from the output of a preliminary run with the values of MAXDPT, LIWORK and LRWORK set to 1. If however only LIWORK and LRWORK are set to 1, then the appropriate values of these parameters for the given value of MAXDPT will be computed and printed unless MAXDPT < 2. In both cases H02BBF will then terminate with IFAIL = 6.

23: IFAIL — INTEGER *Input/Output*

On entry: IFAIL must be set to 0, -1 or 1. Users who are unfamiliar with this parameter should refer to Chapter P01 for details.

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

For this routine, because the values of output parameters may be useful even if IFAIL $\neq 0$ on exit, users are recommended to set IFAIL to -1 before entry. **It is then essential to test the value of IFAIL on exit**.

5.1 Description of Printed Output

The level of printed output from H02BBF is controlled by the user (see the description of MSGLVL in Section 5).

When MSGLVL > 0, the summary printout at the end of execution of H02BBF includes a listing of the status of every variable and constraint. Note that default names are assigned to all variables and constraints. The following describes the printout for each variable.

`Varbl`	gives the name (`V`) and index j, for $j = 1, 2, \ldots, n$ of the variable.
`State`	gives the state of the variable (`FR` if neither bound is in the working set, `EQ` if a fixed variable, `LL` if on its lower bound, `UL` if on its upper bound, `TF` if temporarily fixed at its current value). If `Value` lies outside the upper or lower bounds by more than the feasibility tolerance, `State` will be `++` or `--` respectively.
`Value`	is the value of the variable at the final iteration.
`Lower Bound`	is the lower bound specified for the variable. (`None` indicates that BL(j) $\leq -$BIGBND.) Note that if INTVAR(j) = 1, then the printed value of `Lower Bound` for the jth variable may not be the same as that originally supplied in BL(j).
`Upper Bound`	is the upper bound specified for the variable. (`None` indicates that BU(j) $\geq$ BIGBND.) Note that if INTVAR(j) = 1, then the printed value of `Upper Bound` for the jth variable may not be the same as that originally supplied in BU(j).
`Lagr Mult`	is the value of the Lagrange multiplier for the associated bound constraint. This will be zero if `State` is `FR` or `TF`. If x is optimal, the multiplier should be non-negative if `State` is `LL`, and non-positive if `State` is `UL`.
`Residual`	is the difference between the variable `Value` and the nearer of its bounds BL(j) and BU(j).

The meaning of the printout for general constraints is the same as that given above for variables, except that 'variable' is replaced by 'constraint', BL(j) and BU(j) are replaced by BL($n+j$) and BU($n+j$) respectively, and with the following change in the heading.

`L Con` Gives the name (`L`) and index j, for $j = 1, 2, \ldots, m$ of the constraint.

When MSGLVL > 1, the summary printout at the end of every node during the execution of H02BBF is a listing of the outcome of forcing an integer variable with a non-integer value to take a value within its specified lower and upper bounds.

`Node No` is the current node number of the B&B tree being investigated.

`Parent Node` is the parent node number of the current node.

`Obj Value` is the final objective function value. If a node does not have a feasible solution then `No Feas Soln` is printed instead of the objective function value. If a node whose optimum solution exceeds the best integer solution so far is encountered (i.e., it does not pay to explore the sub-problem any further), then its objective function value is printed together with a `CO` (Cut Off).

`Varbl Chosen` is the index of the integer variable chosen for branching.

`Value Before` is the non-integer value of the integer variable chosen.

`Lower Bound` is the lower bound value that the integer variable is allowed to take.

`Upper Bound` is the upper bound value that the integer variable is allowed to take.

`Value After` is the value of the integer variable after the current optimization.

`Depth` is the depth of the B&B tree at the current node.

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 or warnings specified by the routine:

IFAIL = 1

No feasible integer point was found, i.e., it was not possible to satisfy all the integer variables to within the integer feasibility tolerance (determined by TOLIV). Increase TOLIV and rerun H02BBF.

IFAIL = 2

The original LP solution appears to be unbounded. This value of IFAIL implies that a step as large as BIGBND would have to be taken in order to continue the algorithm (see Section 8).

IFAIL = 3

No feasible point was found for the original LP problem, i.e., it was not possible to satisfy all the constraints to within the feasibility tolerance (determined by TOLFES). If the data for the constraints are accurate only to the absolute precision σ, the user should ensure that the value of the feasibility tolerance is greater than σ. For example, if all elements of A are of order unity and are accurate only to three decimal places, the feasibility tolerance should be at least 10^{-3} (see Section 8).

IFAIL = 4

The maximum number of iterations (determined by ITMAX) was reached before normal termination occurred for the original LP problem (see Section 8).

IFAIL = 5

Not used by this routine.

IFAIL = 6

An input parameter is invalid.

IFAIL = 7

The IP solution reported is not the optimum IP solution. In other words, the B&B tree search for at least one of the branches had to be terminated since an LP sub-problem in the branch did not have a solution (see Section 8).

IFAIL = 8

The maximum depth of the B&B tree used for branch and bound (determined by MAXDPT) is too small. Increase MAXDPT (along with LIWORK and/or LRWORK if appropriate) and rerun H02BBF.

IFAIL = 9

The IP solution reported is the best IP solution for the number of nodes (determined by MAXNOD) investigated in the B&B tree.

IFAIL = 10

No feasible integer point was found for the number of nodes (determined by MAXNOD) investigated in the B&B tree.

Overflow

It may be possible to avoid the difficulty by increasing the magnitude of the feasibility tolerance (TOLFES) and rerunning the program. If the message recurs even after this change, see Section 8.

7 Accuracy

The routine implements a numerically stable active set strategy and returns solutions that are as accurate as the condition of the problem warrants on the machine.

8 Further Comments

The original LP problem may not have an optimum solution, i.e., H02BBF terminates with IFAIL = 2,3,4 or overflow may occur. In this case, the user is recommended to relax the integer restrictions of the problem and try to find the optimum LP solution by using E04MFF instead.

In the B&B method, it is possible for an LP sub-problem to terminate without finding a solution. This may occur due to the number of iterations exceeding the maximum allowed. Therefore the B&B tree search for that particular branch cannot be continued. Thus the final IP solution reported is not the optimum IP solution (IFAIL = 7). For the second and unlikely case, a solution could not be found despite a second attempt at an LP solution.

9 Example

To solve the integer programming problem:

maximize

$$F(x) = 3x_1 + 4x_2$$

subject to the bounds

$$\begin{array}{r} x_1 \geq 0 \\ x_2 \geq 0 \end{array}$$

and to the general constraints

$$\begin{array}{l} 2x_1 + 5x_2 \leq 15 \\ 2x_1 - 2x_2 \leq 5 \\ 3x_1 + 2x_2 \geq 5 \end{array}$$

where x_1 and x_2 are integer variables.

The initial point, which is feasible, is

$$x_0 = (1,1)^T,$$

and $F(x_0) = 7$.

The optimal solution is

$$x^* = (2,2)^T,$$

and $F(x^*) = 14$.

Note that maximizing $F(x)$ is equivalent to minimizing $-F(x)$.

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.

```
*     H02BBF Example Program Text
*     Mark 16 Revised. NAG Copyright 1993.
*     .. Parameters ..
      INTEGER          NIN, NOUT
      PARAMETER        (NIN=5,NOUT=6)
      INTEGER          NMAX, MMAX
      PARAMETER        (NMAX=10,MMAX=10)
      INTEGER          LDA
      PARAMETER        (LDA=MMAX)
      INTEGER          LIWORK, LRWORK
      PARAMETER        (LIWORK=1000,LRWORK=1000)
*     .. Local Scalars ..
      real             BIGBND, OBJMIP, TOLFES, TOLIV
      INTEGER          I, IFAIL, INTFST, ITMAX, J, M, MAXDPT, MAXNOD,
     +                 MSGLVL, N
```

```
*     .. Local Arrays ..
      real              A(LDA,NMAX), BL(MMAX+NMAX), BU(MMAX+NMAX),
     +                  CVEC(NMAX), RWORK(LRWORK), X(NMAX)
      INTEGER           INTVAR(NMAX), IWORK(LIWORK)
*     .. External Subroutines ..
      EXTERNAL          H02BBF
*     .. Executable Statements ..
      WRITE (NOUT,*) 'H02BBF Example Program Results'
*     Skip heading in data file
      READ (NIN,*)
      READ (NIN,*) N, M
      IF (N.LE.NMAX .AND. M.LE.MMAX) THEN
*
*        Read ITMAX, MSGLVL, MAXNOD, INTFST, MAXDPT, TOLFES, TOLIV,
*        CVEC, A, BIGBND, BL, BU, INTVAR and X from data file
*
         READ (NIN,*) ITMAX, MSGLVL
         READ (NIN,*) MAXNOD
         READ (NIN,*) INTFST, MAXDPT
         READ (NIN,*) TOLFES, TOLIV
         READ (NIN,*) (CVEC(I),I=1,N)
         READ (NIN,*) ((A(I,J),J=1,N),I=1,M)
         READ (NIN,*) BIGBND
         READ (NIN,*) (BL(I),I=1,N+M)
         READ (NIN,*) (BU(I),I=1,N+M)
         READ (NIN,*) (INTVAR(I),I=1,N)
         READ (NIN,*) (X(I),I=1,N)
*
*        Solve the IP problem
*
         IFAIL = -1
*
         CALL H02BBF(ITMAX,MSGLVL,N,M,A,LDA,BL,BU,INTVAR,CVEC,MAXNOD,
     +               INTFST,MAXDPT,TOLIV,TOLFES,BIGBND,X,OBJMIP,IWORK,
     +               LIWORK,RWORK,LRWORK,IFAIL)
*
      END IF
      STOP
      END
```

9.2 Program Data

```
H02BBF Example Program Data
  2  3                                           :Values of N and M
  0  10                                          :Values of ITMAX and MSGLVL
  0                                              :Value of MAXNOD
  0  4                                           :Values of INTFST and MAXDPT
  0.0   0.0                                      :Values of TOLFES and TOLIV
 -3.0  -4.0                                      :End of CVEC
  2.0   5.0
  2.0  -2.0
  3.0   2.0                                      :End of matrix A
  1.0E+20                                        :Value of BIGBND
  0.0      0.0      -1.0E+20  -1.0E+20  5.0      :End of BL
  1.0E+20  1.0E+20  15.0       5.0      1.0E+20  :End of BU
  1     1                                        :End of INTVAR
  1.0   1.0                                      :End of X
```

9.3 Program Results

```
H02BBF Example Program Results

*** H02BBF
*** Start of NAG Library implementation details ***

Implementation title: Generalised Base Version
           Precision: FORTRAN double precision
        Product Code: FLBAS17D
                Mark: 17A

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

Parameters
----------

Linear constraints......         3       First integer solution..      OFF
Variables...............         2       Max depth of the tree...        4

Feasibility tolerance...  1.05E-08       Print level.............       10
Infinite bound size.....  1.00E+20       EPS (machine precision).  1.11E-16

Integer feasibility tol.  1.00E-05       Iteration limit.........       50
Max number of nodes.....      NONE

** Workspace provided with MAXDPT =     4: LRWORK =  1000  LIWORK =  1000
** Workspace required with MAXDPT =     4: LRWORK =    84  LIWORK =   137


  *** Optimum LP solution ***   -17.50000


Varbl State     Value     Lower Bound   Upper Bound    Lagr Mult   Residual

V  1    FR    3.92857      0.000000E+00     None       0.0000E+00   3.929
V  2    FR    1.42857      0.000000E+00     None       0.0000E+00   1.429


L Con State     Value     Lower Bound   Upper Bound    Lagr Mult   Residual

L  1    UL    15.0000         None         15.0000     -1.0000      0.0000E+00
L  2    UL    5.00000         None         5.00000     -0.5000     -8.8818E-16
L  3    FR    14.6429       5.00000         None       0.0000E+00   9.643


  *** Start of tree search ***


 Node  Parent    Obj        Varbl  Value       Lower     Upper      Value    Depth
  No    Node    Value      Chosen Before       Bound     Bound      After
   2     1    No Feas Soln    1     3.93       4.00       None       4.00        1
   3     1     -16.2          1     3.93     0.000E+00   3.00       3.00        1
   4     3     -15.5          2     1.80       2.00       None       2.00        2
   5     3     -13.0          2     1.80     0.000E+00   1.00       1.00        2
     *** Integer solution ***
```

```
 Node  Parent    Obj       Varbl  Value      Lower      Upper      Value    Depth
  No    Node    Value      Chosen Before     Bound      Bound      After
   6     4    No Feas Soln    1     2.50      3.00       3.00       3.00       3
   7     4     -14.8          1     2.50    0.000E+00   2.00       2.00       3
   8     7     -12.0     CO   2     2.20      3.00       None       3.00       4
   9     7     -14.0          2     2.20      2.00       2.00       2.00       4
     *** Integer solution ***

    *** End of tree search ***


Total of     9 nodes investigated.

Exit H02BBF - Optimum IP solution found.

Final IP objective value =   -14.00000



Varbl State     Value     Lower Bound   Upper Bound    Lagr Mult   Residual

V   1    UL    2.00000      0.000000E+00   2.00000      -3.000      0.0000E+00
V   2    EQ    2.00000       2.00000       2.00000      -4.000      0.0000E+00


L Con State     Value     Lower Bound   Upper Bound    Lagr Mult   Residual

L   1    FR    14.0000        None         15.0000      0.0000E+00   1.000
L   2    FR   0.000000E+00    None         5.00000      0.0000E+00   5.000
L   3    FR    10.0000       5.00000         None       0.0000E+00   5.000
```