

NAG Library Routine Document

H02BBF

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

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, CVEC,      &
                  MAXNOD, INTFST, MAXDPT, TOLIV, TOLFES, BIGBND, X,      &
                  OBJMIP, IWORK, LIWORK, RWORK, LRWORK, IFAIL)
INTEGER           ITMAX, MSGLVL, N, M, LDA, INTVAR(N), MAXNOD, INTFST,    &
                  MAXDPT, IWORK(LIWORK), LIWORK, LRWORK, IFAIL
REAL (KIND=nag_wp) A(LDA,*), BL(N+M), BU(N+M), CVEC(N), TOLIV, 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 (1987). 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

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

subject to

$$\sum_{j=1}^n a_{ij}x_j \begin{cases} = \\ \leq \\ \geq \end{cases} b_i, \quad i = 1, 2, \dots, m$$

$$l_j \leq x_j \leq u_j, \quad j = 1, 2, \dots, n \quad (1)$$

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 (1965) or Mitra (1973)) 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 noninteger 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/E04MFA.

4 References

- Dakin R J (1965) A tree search algorithm for mixed integer programming problems *Comput. J.* **8** 250–255
- Mitra G (1973) Investigation of some branch and bound strategies for the solution of mixed integer linear programs *Math. Programming* **4** 155–170
- Taha H A (1987) *Operations Research: An Introduction* Macmillan, New York

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: MSGVLVL – 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 (KIND=nag_wp) array *Input*
Note: the second dimension of the array A must be at least N if $M > 0$ and at least 1 if $M = 0$.
On entry: the i th row of A must contain the coefficients of the i th general constraint, for $i = 1, 2, \dots, m$.
 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 (KIND=nag_wp) array *Input*
 8: BU(N + M) – REAL (KIND=nag_wp) 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 nonexistent lower bound (i.e., $l_j = -\infty$), set $BL(j) \leq -BIGBND$ and to specify a nonexistent upper bound (i.e., $u_j = +\infty$), set $BU(j) \geq BIGBND$. To specify the j th constraint as an equality, set $BL(j) = BU(j) = \beta$, say, where $|\beta| < BIGBND$.

Constraints:

$$BL(j) \leq BU(j), \text{ for } j = 1, 2, \dots, N + M;$$

$$\text{if } BL(j) = BU(j) = \beta, |\beta| < BIGBND.$$

- 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:

$$\text{INTVAR}(j) = 0 \text{ or } 1, \text{ for } j = 1, 2, \dots, N;$$

$$\text{INTVAR}(j) = 1 \text{ for at least one value of } j.$$
- 10: CVEC(N) – REAL (KIND=nag_wp) array *Input*
On entry: the coefficients c_j of the objective function $F(x) = c_1x_1 + c_2x_2 + \dots + c_nx_n$. 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 $j = 1, 2, \dots, n$, 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 $\text{MAXNOD} \leq 0$ and $\text{INTFST} \leq 0$, 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 $\text{INTFST} > 0$ then the B&B tree search is terminated at node k say, which contains the first integer solution. For $\text{MAXNOD} > 0$ this applies only if $k \leq \text{MAXNOD}$.
- 13: MAXDPT – INTEGER *Input*
On entry: the maximum depth of the B&B tree used for branch and bound.
Suggested value: $\text{MAXDPT} = 3 \times N/2$.
Constraint: $\text{MAXDPT} \geq 2$.
- 14: TOLIV – REAL (KIND=nag_wp) *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 near unity then x_j is considered to be integer only if $(1 - \text{TOLIV}) \leq x_j \leq (1 + \text{TOLIV})$.
On exit: unchanged if on entry $\text{TOLIV} > 0.0$, else $\text{TOLIV} = 10^{-5}$.
- 15: TOLFES – REAL (KIND=nag_wp) *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 $\text{TOLFES} > 0.0$, else $\text{TOLFES} = \sqrt{\epsilon}$ (where ϵ is the *machine precision*).
- 16: BIGBND – REAL (KIND=nag_wp) *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 $-\text{BIGBND}$ will be regarded as $-\infty$.
On exit: unchanged if on entry $\text{BIGBND} > 0.0$, else $\text{BIGBND} = 10^{20}$.
- 17: X(N) – REAL (KIND=nag_wp) 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 (KIND=nag_wp) 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 Communication Array
20: LIWORK – INTEGER Input

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

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

21: RWORK(LRWORK) – REAL (KIND=nag_wp) array Communication Array
22: LRWORK – INTEGER Input

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

Constraint: $LRWORK \geq MAXDPT \times (N + 1) + 2 \times \min(N, M + 1)^2 + 14 \times N + 12 \times M$.

If MSGGLVL > 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, you 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. If you are unfamiliar with this parameter you should refer to Section 3.3 in the Essential Introduction for details.

For environments where it might be inappropriate to halt program execution when an error is detected, the value -1 or 1 is recommended. If the output of error messages is undesirable, then the value 1 is recommended. Otherwise, because for this routine the values of the output parameters may be useful even if IFAIL \neq 0 on exit, the recommended value is -1. **When the value -1 or 1 is used it is essential to test the value of IFAIL on exit.**

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

5.1 Description of Printed Output

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

When MSGGLVL > 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 $j = 1, 2, \dots, n$, 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 iterate.

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 j th 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 j th 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, with ‘variable’ 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 $j = 1, 2, \dots, m$, of the constraint.

When $MSGVLV > 1$, 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 noninteger 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 noninteger 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).

Note: H02BBF may return useful information for one or more of the following detected errors or warnings.

Errors or warnings detected 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 , you 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.

IFAIL = 11

Although the workspace sizes are sufficient to meet the documented restriction, they are not sufficiently large to accommodate an internal partition of the workspace that meets the requirements of the problem. Increase the workspace sizes.

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

H02BBF 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 or 4 or overflow may occur. In this case, you are recommended to relax the integer restrictions of the problem and try to find the optimum LP solution by using E04MFF/E04MFA 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 returned solution may not be optimal. (IFAIL = 7). For the second and unlikely case, a solution could not be found despite a second attempt at an LP solution.

9 Example

This example solves the integer programming problem:

maximize

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

subject to the bounds

$$\begin{aligned} x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

and to the general constraints

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

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

```

Program h02bbfe

!      H02BBF Example Program Text

!      Mark 24 Release. NAG Copyright 2012.

!      .. Use Statements ..
Use nag_library, Only: h02bbf, nag_wp
!      .. Implicit None Statement ..
Implicit None
!      .. Parameters ..
Integer, Parameter          :: nin = 5, nout = 6
!      .. Local Scalars ..
Real (Kind=nag_wp)         :: bigbnd, objmip, tolfes, toliv
Integer                    :: i, ifail, intfst, itmax, j, lda,      &
                             liwork, lrwork, m, maxdpt, maxnod,    &
                             msglvl, n
!      .. Local Arrays ..
Real (Kind=nag_wp), Allocatable :: a(:, :), bl(:), bu(:), cvec(:),    &
                             rwork(:), x(:)

```

```

Integer, Allocatable      :: intvar(:), iwork(:)
! .. Intrinsic Procedures ..
Intrinsic                 :: min
! .. Executable Statements ..
Write (nout,*) 'H02BBF Example Program Results'
Flush (nout)

! Skip heading in data file
Read (nin,*)

Read (nin,*) n, m
lda = m
Allocate (a(lda,n),bl(m+n),bu(m+n),cvec(n),x(n),intvar(n))

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)

liwork = (25+n+m)*maxdpt + 5*n + m + 4
lrwork = maxdpt*(n+1) + 2*min(n,m+1)**2 + 14*n + 12*m
Allocate (iwork(liwork),rwork(lrwork))

! Solve the IP problem

ifail = 0
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 Program h02bbfe

```

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

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 = 84 LIWORK = 137
 ** 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	0.00000	None	0.000	3.929
V 2	FR	1.42857	0.00000	None	0.000	1.429

L Con	State	Value	Lower Bound	Upper Bound	Lagr Mult	Residual
L 1	UL	15.0000	None	15.0000	-1.000	0.000
L 2	UL	5.00000	None	5.00000	-0.5000	-8.8818E-16
L 3	FR	14.6429	5.00000	None	0.000	9.643

*** Start of tree search ***

Node No	Parent Node	Obj Value	Varbl Chosen	Value Before	Lower Bound	Upper Bound	Value After	Depth
2	1	No Feas Soln	1	3.93	4.00	None	4.00	1
3	1	-16.2	1	3.93	0.00	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	0.00	1.00	1.00	2

*** Integer solution ***

Node No	Parent Node	Obj Value	Varbl Chosen	Value Before	Lower Bound	Upper Bound	Value After	Depth
6	4	No Feas Soln	1	2.50	3.00	3.00	3.00	3
7	4	-14.8	1	2.50	0.00	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	0.00000	2.00000	-3.000	0.000
V 2	EQ	2.00000	2.00000	2.00000	-4.000	0.000

L Con	State	Value	Lower Bound	Upper Bound	Lagr Mult	Residual
L 1	FR	14.0000	None	15.0000	0.000	1.000
L 2	FR	0.00000	None	5.00000	0.000	5.000
L 3	FR	10.0000	5.00000	None	0.000	5.000