

NAG Toolbox for Matlab Chapter Introduction

H – Operations Research

Contents

1	Scope of the Chapter	2
2	Background to the Problems	2
3	Recommendations on Choice and Use of Available Functions	4
4	Index	5
5	References	5

1 Scope of the Chapter

This chapter provides functions to solve certain integer programming, transportation and shortest path problems.

2 Background to the Problems

General **linear programming** (LP) problems (see Dantzig (1963)) are of the form:

$$\text{find } x = (x_1, x_2, \dots, x_n)^T \text{ to maximize } F(x) = \sum_{j=1}^n c_j x_j$$

subject to linear constraints which may have the forms:

$$\sum_{j=1}^n a_{ij} x_j = b_i, \quad i = 1, 2, \dots, m_1 \quad (\text{equality})$$

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = m_1 + 1, \dots, m_2 \quad (\text{inequality})$$

$$\sum_{j=1}^n a_{ij} x_j \geq b_i, \quad i = m_2 + 1, \dots, m \quad (\text{inequality})$$

$$x_j \geq l_j, \quad j = 1, 2, \dots, n \quad (\text{simple bound})$$

$$x_j \leq u_j, \quad j = 1, 2, \dots, n \quad (\text{simple bound})$$

This chapter deals with **integer programming** (IP) problems in which some or all the elements of the solution vector x are further constrained to be **integers**. For general LP problems where x takes only real (i.e., noninteger) values, refer to Chapter E04.

IP problems may or may not have a solution, which may or may not be unique.

Consider for example the following problem:

$$\begin{aligned} & \text{minimize} && 3x_1 + 2x_2 \\ & \text{subject to} && 4x_1 + 2x_2 \geq 5 \\ & && 2x_2 \leq 5 \\ & \text{and} && x_1 - x_2 \leq 2 \\ & && x_1 \geq 0, x_2 \geq 0. \end{aligned}$$

The hatched area in Figure 1 is the **feasible region**, the region where all the constraints are satisfied, and the points within it which have integer co-ordinates are circled. The lines of hatching are in fact contours of decreasing values of the objective function $3x_1 + 2x_2$, and it is clear from Figure 1 that the optimum IP solution is at the point (1, 1). For this problem the solution is unique.

However, there are other possible situations.

- (a) There may be more than one solution; e.g., if the objective function in the above problem were changed to $x_1 + x_2$, both (1, 1) and (2, 0) would be IP solutions.
- (b) The feasible region may contain no points with integer co-ordinates, e.g., if an additional constraint

$$3x_1 \leq 2$$

were added to the above problem.

- (c) There may be no feasible region, e.g., if an additional constraint

$$x_1 + x_2 \leq 1$$

were added to the above problem.

- (d) The objective function may have no finite minimum within the feasible region; this means that the feasible region is unbounded in the direction of decreasing values of the objective function, e.g., if the constraints

$$4x_1 + 2x_2 \geq 5, \quad x_1 \geq 0, \quad x_2 \geq 0,$$

were deleted from the above problem.

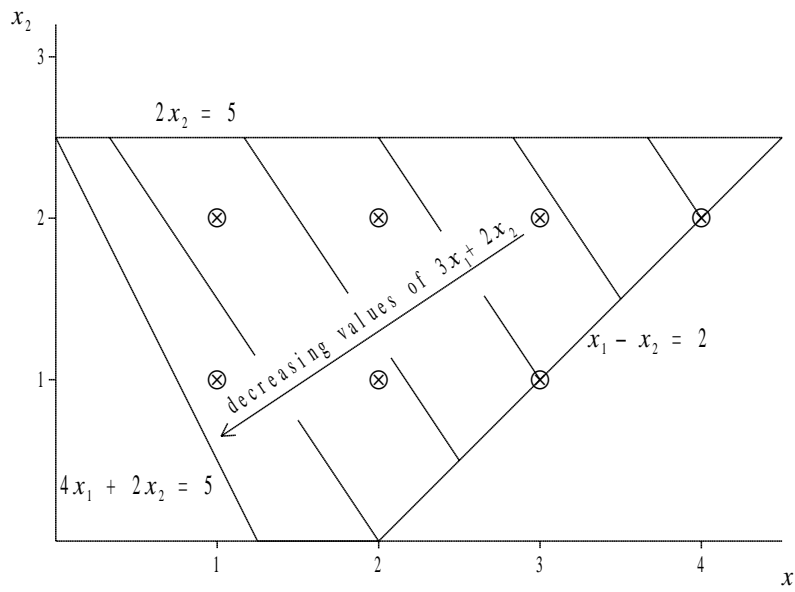


Figure 1

Algorithms for IP problems are usually based on algorithms for general LP problems, together with some procedure for constructing additional constraints which exclude noninteger solutions (see Beale (1977)).

The Branch and Bound (B&B) method is a well-known and widely used technique for solving IP problems (see Beale (1977) or Mitra (1973)). It involves subdividing the optimum solution to the original LP problem into two mutually exclusive sub-problems by branching an integer variable that currently has a fractional optimal value. Each sub-problem can now be solved as an LP problem, using the objective function of the original problem. The process of branching continues until a solution for one of the sub-problems is feasible with respect to the integer problem. In order to prove the optimality of this solution, the rest of the sub-problems in the B&B tree must also be solved. Naturally, if a better integer feasible solution is found for any sub-problem, it should replace the one at hand.

The efficiency in computations is enhanced by discarding inferior sub-problems. These are problems in the B&B search tree whose LP solutions are lower than (in the case of maximization) the best integer solution at hand.

The B&B method may also be applied to convex quadratic programming (QP) problems. Routines have been introduced into this chapter to formally apply the technique to dense general QP problems and to sparse LP or QP problems.

A special type of linear programming problem is the **transportation** problem in which there are $p \times q$ variables y_{kl} which represent quantities of goods to be transported from each of p sources to each of q destinations.

The problem is to minimize

$$\sum_{k=1}^p \sum_{l=1}^q c_{kl} y_{kl}$$

where c_{kl} is the unit cost of transporting from source k to destination l . The constraints are:

$$\begin{aligned}\sum_{l=1}^q y_{kl} &= A_k \quad (\text{availabilities}) \\ \sum_{k=1}^p y_{kl} &= B_l \quad (\text{requirements}) \\ y_{kl} &\geq 0.\end{aligned}$$

Note that the availabilities must equal the requirements:

$$\sum_{k=1}^p A_k = \sum_{l=1}^q B_l = \sum_{k=1}^p \sum_{l=1}^q y_{kl}$$

and if all the A_k and B_l are integers, then so are the optimal y_{kl} .

The **shortest path** problem is that of finding a path of minimum length between two distinct vertices n_s and n_e through a network. Suppose the vertices in the network are labelled by the integers $1, 2, \dots, n$. Let (i, j) denote an ordered pair of vertices in the network (where i is the origin vertex and j the destination vertex of the arc), x_{ij} the amount of flow in arc (i, j) and d_{ij} the length of the arc (i, j) . The LP formulation of the problem is thus given as

$$\text{minimize} \quad \sum \sum d_{ij} x_{ij} \text{ subject to } Ax = b, \quad 0 \leq x \leq 1, \quad (1)$$

where

$$a_{ij} = \begin{cases} +1 & \text{if arc } j \text{ is directed away from vertex } i, \\ -1 & \text{if arc } j \text{ is directed towards vertex } i, \\ 0 & \text{otherwise} \end{cases}$$

and

$$b_i = \begin{cases} +1 & \text{for } i = n_s, \\ -1 & \text{for } i = n_e, \\ 0 & \text{otherwise.} \end{cases}$$

The above formulation only yields a meaningful solution if $x_{ij} = 0$ or 1 ; that is, arc (i, j) forms part of the shortest route only if $x_{ij} = 1$. In fact since the optimal LP solution will (in theory) always yield $x_{ij} = 0$ or 1 , (1) can also be solved as an IP problem. Note that the problem may also be solved directly (and more efficiently) using a variant of Dijkstra's algorithm (see Ahuja *et al.* (1993)).

The **travelling salesman** problem is that of finding a minimum distance route round a given set of cities. The salesperson must visit each city only once before returning to his or her city of origin. It can be formulated as an IP problem in a number of ways. One such formulation is described in Williams (1993). There are currently no functions in the Library for solving such problems.

3 Recommendations on Choice and Use of Available Functions

h02bb solves dense integer programming problems using a branch and bound method.

h02bv prints the solution to an integer or a linear programming problem using specified names for rows and columns.

h02bz supplies further information on the optimum solution obtained by h02bb.

h02cb solves dense integer general quadratic programming problems.

h02cd supplies optional parameter values to h02cb.

h02ce solves sparse integer linear programming or quadratic programming problems.

h02cg supplies optional parameter values to h02ce.

h03ab solves transportation problems. It uses integer arithmetic throughout and so produces exact results. On a few machines, however, there is a risk of integer overflow without warning, so the integer values in the data should be kept as small as possible by dividing out any common factors from the coefficients of the constraint or objective functions.

h03ad solves shortest path problems using Dijkstra's algorithm.

h02bb and h03ab treat all matrices as dense and hence are not intended for large sparse problems. For solving large sparse LP problems, use e04nq or e04ug.

4 Index

Convert data to arrays for use with h02bb or e04mf	h02bu
Integer programming problem (dense):	
print solution with specified names	h02bv
solve LP problem using branch and bound method	h02bb
solve QP problem using branch and bound method	h02cb
supply further information on the solution obtained from h02bb	h02bz
Integer programming problem (sparse):	
solve LP or QP problem using branch and bound method using e04nk	h02ce
MPSX data input, defining IP or LP problem:	
interpret data, optimize and print solution	h02bf
Read optional parameter values from external file for h02cb	h02cc
Read optional parameter values from external file for h02ce	h02cf
Shortest path, through directed or undirected network	h03ad
Supply optional parameter values to h02cb	h02cd
Supply optional parameter values to h02ce	h02cg
Transportation problem	h03ab

5 References

- Ahuja R K, Magnanti T L and Orlin J B (1993) *Network Flows: Theory, Algorithms and Applications* Prentice-Hall
- Beale E M (1977) Integer Programming *The State of the Art in Numerical Analysis* (ed D A H Jacobs) Academic Press
- Dantzig G B (1963) *Linear Programming and Extensions* Princeton University Press
- IBM (1971) MPSX – Mathematical programming system *Program Number 5734 XM4* IBM Trade Corporation, New York
- Mitra G (1973) Investigation of some branch and bound strategies for the solution of mixed integer linear programs *Math. Programming* **4** 155–170
- Williams H P (1993) *Model Building in Mathematical Programming* (3rd Edition) Wiley