Solution of Integer Programs with a Quadratic Objective Function

TA HUU PHUONG

London School of Economics, London, England

ABSTRACT. This paper presents a branch and bound method for solving problems in which the objective function is quadratic, the constraints are linear, and some or all variables are required to be integer. The algorithm is obtained by grafting an inverse-basis version of Beale's method onto the Land-Doig procedure The code has been tested on a computer, and computational results with various strategies of branching are reported.

KEY WORDS AND PHRASES: algorithm, branch and bound, integer programming, quadratic programming, quasiconcavity

CR CATEGORIES 5 41

1. *Introduction*

A number of methods [1–3, 7, 11, 13] have been proposed for the solution of discrete quadratic programming problems. Little, however, has been done on the implementation of computer routines. The purpose of this paper is to provide an algorithm for solving quadratic programs which are "well-behaved," but where some or all variables are required to be integer. By well-behaved we mean Quadratic Programming (QP) problems which are valid for submission to a quasiconcave QP maximizing routine.[1] As limited by such a restriction the algorithm should not, therefore, be used for the solution of such problems as the quadratic assignment.

The code is obtained by grafting (with some modifications) an inverse-basis version of Beale's method onto a branch and bound procedure of the Land-Doig type. Computational experience with various strategies of branching is discussed.

2. *Problem and Method*

The discrete quadratic programming problem we consider can be written in the following form:

$$\text{(P)}\quad \begin{aligned} \text{Maximize}\quad & f = p'x + \tfrac{1}{2}x'Dx \\ \text{subject to}\quad & Ax \leq b, \\ & x \geq 0, \\ & x_j \text{ integer},\ j \in N_1, \end{aligned}$$

where D is a symmetric matrix of order n, b, p are $m \times 1$ and $1 \times n$ vectors, respectively; N_1 is a subset of $N = (1, 2, \cdots, n)$; and x is the variable vector to be determined.

In the absence of the integrality condition we refer to (P) as the relaxed problem.

Author's present address School of Accountancy and Business Administration, University of Singapore, Bukit Timah Road, Singapore 10, Singapore

[1] Some of those routines can be found in [15] For definitions of different classes of concavity and their properties, see, for instance, [6, 12, 17].

We shall also restrict our discussion to the case where the objective function is quasiconcave on the nonnegative orthant. This restriction is needed to ensure a global optimum to the relaxed problem and the convergence of the algorithm.

The method is based essentially on the approach of Land and Doig [8]. That is, one first obtains the solution of the relaxed problem. If it is nonintegral, an integer-constrained variable is then fixed at its nearest integral value. Of course, here one must use a QP procedure to solve the successive QP subproblems which arise from the branching and backtracking operations. The QP procedure employed is that described by Land and Morton [9]. Its adoption is prompted by the following considerations:

1. As a version of Beale's method, the procedure is capable of solving QP problems having quasiconcave objective function.

2. Apart from other attractive features such as economy of storage space, accuracy, and efficiency, the inverse-basis structure of the procedure also

(i) facilitates the addition and removal of constraints whenever the need demands;

(ii) allows the use of dual simplex criteria to restore the feasibility of an integer-constrained variable when it is forced to take an integral value; and

(iii) provides sufficient information at an optimal tableau which enables us to specify rules for choosing a branching variable and a branching direction.

As for the bounding, one could use these facilities to obtain the bounds of the unexplored nodes. If one is not prepared to go further than the information given at an optimal tableau, one could estimate these bounds from the rates of fall of the (linear) tangential function at a QP optimal solution. Unfortunately, this way of estimating bounds can only be done when the maximizing function is concave. When the function is nonconcave, exploration of the nodes requires explicit solutions of the QP subproblems.

3. *Computational Results*

The algorithm has been tested on five sets of problems having concave objective function. Matrices A and D and the solutions of the relaxed problems are generated using a random generator. Other components are formed according to the scheme reported by Rosen and Suzuki [18]. Each set consists of ten problems having from 5 to 20 quadratic discrete variables. In order to test the mechanism of the auxiliary constraints (i.e. constraints that are added in order to form artificial corner points) on the algorithm as a whole and on the branching process in particular, we decide to limit the number of (original) constraints of each problem to two, one of which will be effective at the QP optimal solution.

The following variants with respect to different strategies on the selection of the branching variable and branching direction have been tested on each problem.

Variant 1. The variable which yields the greatest reduction in the linear tangential function at an optimal tableau is chosen as the branching variable (the greatest cost rule). The adjacent integer value corresponding to the least reduction in the functional value is chosen as the branching direction.

Variant 2. Choose the first integer-constrained variable which appears in the basic optimal tableau but does not satisfy the integrality condition as the branching variable (the lexicographic rule). The branching direction is similar to that in Variant 1.

Variant 3. Select from among the integer-constrained variables the variable which has the largest fractional part as the branching variable (the greatest fractional rule). The direction is similar to that in Variant 1.

Variant 4. Both branching variable and branching direction are selected in the same way as that for Variant 1, except that when the costs in going to the neighboring integer values are equal, the direction corresponding to the nearest integer is selected.

Variant 5. Follow the same rules in Variant 2, with the additional proviso described in Variant 4.

Variant 6. Use Variant 3 together with the additional proviso described in Variant 4.

TABLE I

Set	Variant	First discrete solution	Optimal discrete solution	Proof of optimality	Discrepancy between first and optimal value	Time
		No. of iterations	No. of iterations	No. of iterations	%	Seconds
1	1	20	78	105	13.30	.439
	2	19	82	109	14.00	.438
	3	20	83	121	15.25	.446
	4	21	50	87	5.33	.430
	5	21	51	90	5.47	.428
	6	23	44	95	2.78	.436
2	1	20	72	100	12.83	.434
	2	20	73	99	13.13	.431
	3	20	70	110	14.99	.438
	4	20	46	81	4.86	.426
	5	20	46	81	4.80	.423
	6	23	42	86	4.24	.430
3	1	61	1012	1417	10.51	2.10
	2	61	983	1455	10.78	2.04
	3	60	1676	2061	10.49	2.57
	4	58	676	1149	3.22	1.84
	5	58	692	1147	4.12	1.71
	6	56	304	884	2.48	1.44
4	1	32	4133	7785	17.21	6.46
	2	31	4283	8842	16.33	6.95
	3	33	3354	8850	25.62	7.66
	4	32	750	4855	14.80	4.23
	5	32	2177	6413	14.73	5.43
	6	31	699	6237	11.33	5.61
5	1	27	222	222	2.64	.99
	2	26	207	207	2.46	.97
	3	30	215	215	3.37	.98
	4	26	266	266	5.21	.96
	5	26	315	315	5.40	1.03
	6	31	237	237	2.07	1.00

The performance of each set of problems with respect to different variants is recorded in Table I. There are five quadratic variables in Sets 1 and 2, ten in Set 3, and twenty in Set 5. Set 4 has ten variables, five of which are quadratic. The first column in the table indicates the set number, followed by the variant number in the second column. The number of simplex iterations should provide a reasonable norm for comparing various rules of selection, because it is independent of the machines used. The last column indicates the average computer time spent on the CDC 7600 for the solution of a problem which does not exceed 20,000 iterations to prove optimality.

There is no doubt that Variants 4, 5, and 6 perform much better than Variants 1, 2, and 3 both in the number of iterations and in time. It is fair to say that although occasionally Variant 6 needs more iterations, and therefore more time than Variants 4 and 5, it should be preferable because it produces a good first discrete solution. For large problems (for instance in Set 5) Variant 4 is perhaps the most economical version.

As it is not possible to estimate the rate of reduction of a nonconcave function by its tangential function, the following variant is suggested for a nonconcave quasiconcave program.

Variant 7. Select a branching variable by lexicographic criterion. Choose the integer value nearest to the current fractional value of the variable as the branching direction. Use the functional value at an optimal QP as upper bound to the unexplored nodes or solve the subproblems explicitly.

The algorithm has been implemented on a computer. The program is written in Fortran IV and consists of a main routine and 27 subroutines. We have borrowed directly from the Land-Powell package [10] 17 subroutines and modified another 10, some substantially, some only with minor alterations Detailed description of the modifications can be found in [16]. The data of a test problem from each set and those of a mixed zero-one program are recorded in Appendix A.

4. *Conclusion*

As with any algorithmic procedure, the method has its own advantages and limitations. Perhaps the main advantage of branch and bound is the ease and relatively low cost with which a good integer solution can be obtained. This is particularly apparent in Variant 6.

Another facility offered by the method is the possibility of solving problems in which some or all variables are required to satisfy a certain step size value, in addition to the integrality condition. Occasionally, with the view of simplifying the input to the computer, this facility could and should be used to scale down large coefficients of matrices A or D, or vector p to reasonable magnitude. Moreover, in order to minimize rounding-off errors it is very useful to perform the scaling process in cases where there are great discrepancies in the coefficients.

Needless to say, the method can be applied to solve zero-one problems without any modifications.

A serious drawback of the method lies in the number of branches to be explored which cannot be determined a priori. Particularly with problems of a combinatorial nature, the branches would proliferate so enormously that it is prohibitive, in terms of computer time, to reach the discrete optimal solution.

The method is also limited by the quasiconcavity of the quadratic objective function. If the function is concave, any of the first six variants can be used. If it is nonconcave but quasiconcave, only Variant 7 is valid A computational scheme to help in recognizing the different kinds of concavity is described in Appendix B. What one needs is simply to have access to a routine which computes the eigenvalues of a real square symmetric matrix.

Appendix A. *Some Test Problems*

Problem 1.

Maximize $90.5x_1 + 30.5x_2 - 6.8x_3 + 79.3x_4 + 12.2x_5 - 5x_1^2 - x_1x_2 - 2x_1x_3 - 3x_1x_4 - 4x_1x_5 - 4.5x_2^2 + x_2x_3 - 2x_2x_4 + 3x_2x_5 - 3.5x_3^2 - 3x_3x_4 + 5x_3x_5 - 6x_4^2 - x_4x_5 - 7.5x_5^2$

subject to $-23x_1 + 19x_2 + 26x_3 + 17x_4 - 35x_5 \leq 18$, $41x_1 - 10x_3 + 17x_4 - 38x_5 \leq -99.9$, $x_i \geq 0$, integer $i = 1, \cdots, 5$.

Solution: $x = (1, 4, 2, 4, 5)$.

Problem 2.

Maximize $71.1x_1 - 39.7x_2 - 9.4x_3 + 36.8x_4 + 96.4x_5 - 5x_1^2 - x_1x_2 - 2x_1x_3 - 3x_1x_4 - 4x_1x_5 - 4.5x_2^2 + x_2x_3 - 2x_2x_4 + 3x_2x_5 - 3.5x_3^2 - 3x_3x_4 + 5x_3x_5 - 6x_4^2 - x_4x_5 - 7.5x_5^2$

subject to $27x_1 + 43x_2 + 10x_3 + 32x_4 - 44x_5 \leq 8$; $8x_1 - 38x_2 - 20x_3 + 4x_4 + 35x_5 \leq 57.7$; $x_i \geq 0$, integer $i = 1, \cdots, 5$.

Solution: $x = (2, 1, 5, 1, 5)$.

Problem 3.

Maximize $29.6x_1 + 83x_2 - 20.9x_3 + 106.7x_4 - 3.7x_5 + 78.1x_6 - 7.9x_7 + 63.5x_8 + 61.1x_9 + 24.3x_{10} - 5x_1^2 - 4.5x_2^2 - 3.5x_3^2 - 6x_4^2 - 7.5x_5^2 - 5x_6^2 - 4.5x_7^2 - 3.5x_8^2 - 6x_9^2 - 7.5x_{10}^2 - x_1x_2 - 2x_1x_3 - 3x_1x_4 - 4x_1x_5$

$+ x_2x_3 - 2x_3x_4 + 3x_3x_5 - 3x_3x_4 + 5x_3x_5 - x_4x_5 - x_6x_7 - 2x_6x_8$
$- 3x_6x_9 - 4x_6x_{10} + x_7x_8 - 2x_7x_9 + 3x_7x_{10} - 3x_8x_9 + 5x_8x_{10} - x_9x_{10}$

subject to $11x_1 - 41x_2 + 25x_3 - 12x_4 + 11x_5 - 46x_6 + 47x_7 - 22x_8 + 20x_9 - 28x_{10} \leq 175$; $-16x_1 + 44x_2 - 36x_3 + 42x_4 - 18x_5 + 44x_6 - 18x_7 + 19x_8 - 4x_9 + 39x_{10} \leq 327$; $0 \leq x_i \leq 10$, integer.

Solution: $x = (2, 4, 2, 4, 2, 1, 0, 5, 4, 0)$.

Problem 4.

Maximize $54.3x_1 - 36.5x_2 + 27.6x_3 - 5.2x_4 + 47.8x_5 - 13x_6 + 6x_7 - 44x_8 + 39x_9 - 5x_1^2 - 4.5x_2^2 - 3.5x_3^2 - 6x_4^2 - 7.5x_5^2 - x_1x_2 - 2x_1x_3 - 3x_1x_4 - 4x_1x_5 + x_2x_3 - 2x_3x_4 + 3x_3x_5 - 3x_3x_4 + 5x_3x_5 - x_4x_5$

subject to $22x_1 - 39x_2 + 15x_3 - 37x_4 + 14x_5 - 13x_6 + 6x_7 - 44x_8 + 39x_9 \leq 89$; $-11x_1 + 41x_2 - 25x_3 + 12x_4 - 11x_5 + 46x_6 - 47x_7 + 22x_8 - 20x_9 + 28x_{10} \leq 77$; $0 \leq x_i \leq 10$, integer.

Solution: $x = (0, 2, 5, 1, 4, 2, 10, 0, 1, 0)$.

Problem 5.

Maximize $85x_1 - 12.5x_2 - 77.5x_3 + 64x_4 + 151x_5 - 53.5x_6 + 130.5x_7 + 150.5x_8 - 167x_9 + 54.5x_{10} + 99.5x_{11} - 73.5x_{12} + 93x_{13} + 61.5x_{14} - 133.5x_{15} + 91.5x_{16} + 70x_{17} - 54x_{18} + 35.5x_{19} + 15x_{20} - 2x_1^2 - 0.5x_2^2 - 4.5x_3^2 - 2x_4^2 - 8x_5^2 - 4.5x_6^2 - 4.5x_7^2 - 12.5x_8^2 - 8x_9^2 - 0.5x_{10}^2 - 4.5x_{11}^2 - 0.5x_{12}^2 - 2x_{13}^2 - 0.5x_{14}^2 - 4.5x_{15}^2 - 4.5x_{16}^2 - 2x_{17}^2 - 2x_{18}^2 - 0.5x_{19}^2 - 2x_{20} - 2x_1x_2 + 6x_1x_3 - 4x_1x_4 - 8x_1x_5 + 6x_1x_6 - 6x_1x_7 - 10x_1x_8 + 8x_1x_9 - 2x_1x_{10} - 6x_1x_{11} + 2x_1x_{12} - 4x_1x_{13} - 2x_1x_{14} + 6x_1x_{15} - 6x_1x_{16} - 4x_1x_{17} + 4x_1x_{18} - 2x_1x_{19} - 4x_1x_{20} + 3x_2x_3 - 2x_2x_4 - 4x_2x_5 + 3x_2x_6 - 3x_2x_7 - 5x_2x_8 + 4x_2x_9 - x_2x_{10} - 3x_2x_{11} + x_2x_{12} - 2x_2x_{13} - x_2x_{14} + 3x_2x_{15} - 3x_2x_{16} - 2x_2x_{17} + 2x_2x_{18} - x_2x_{19} - 2x_2x_{20} + 6x_3x_4 + 12x_3x_5 - 9x_3x_6 + 9x_3x_7 + 15x_3x_8 - 12x_3x_9 + 3x_3x_{10} + 9x_3x_{11} - 3x_3x_{12} + 6x_3x_{13} + 3x_3x_{14} - 9x_3x_{15} + 9x_3x_{16} + 6x_3x_{17} - 6x_3x_{18} + 3x_3x_{19} + 6x_3x_{20} - 8x_4x_5 + 6x_4x_6 - 6x_4x_7 - 10x_4x_8 + 8x_4x_9 - 2x_4x_{10} - 6x_4x_{11} + 2x_4x_{12} - 4x_4x_{13} - 2x_4x_{14} + 6x_4x_{15} - 6x_4x_{16} - 4x_4x_{17} + 4x_4x_{18} - 2x_4x_{19} - 4x_4x_{20} + 12x_5x_6 - 12x_5x_7 - 20x_5x_8 + 16x_5x_9 - 4x_5x_{10} - 12x_5x_{11} + 4x_5x_{12} - 8x_5x_{13} - 4x_5x_{14} + 12x_5x_{15} - 12x_5x_{16} + 8x_5x_{17} + 8x_5x_{18} - 4x_5x_{19} - 8x_5x_{20} + 9x_6x_7 + 15x_6x_8 - 12x_6x_9 + 3x_6x_{10} + 9x_6x_{11} - 3x_6x_{12} + 6x_6x_{13} + 3x_6x_{14} - 9x_6x_{15} + 9x_6x_{16} + 6x_6x_{17} - 6x_6x_{18} + 3x_6x_{19} + 6x_6x_{20} - 15x_7x_8 + 12x_7x_9 - 3x_7x_{10} - 9x_7x_{11} + 3x_7x_{12} - 6x_7x_{13} - 3x_7x_{14} + 9x_7x_{15} - 9x_7x_{16} - 6x_7x_{17} + 6x_7x_{18} - 3x_7x_{19} - 6x_7x_{20} + 20x_8x_9 - 5x_8x_{10} - 15x_8x_{11} + 5x_8x_{12} - 10x_8x_{13} - 5x_8x_{14} + 15x_8x_{15} - 15x_8x_{16} - 10x_8x_{17} + 10x_8x_{18} - 5x_8x_{19} - 10x_8x_{20} + 4x_9x_{10} + 12x_9x_{11} - 4x_9x_{12} + 8x_9x_{13} + 4x_9x_{14} - 12x_9x_{15} + 12x_9x_{16} + 8x_9x_{17} - 8x_9x_{18} + 4x_9x_{19} + 8x_9x_{20} - 3x_{10}x_{11} + x_{10}x_{12} - 2x_{10}x_{13} - x_{10}x_{14} + 3x_{10}x_{15} - 3x_{10}x_{16} - 2x_{10}x_{17} + 2x_{10}x_{18} - x_{10}x_{19} - 2x_{10}x_{20} + 3x_{11}x_{12} - 6x_{11}x_{13} - 3x_{11}x_{14} + 9x_{11}x_{15} - 9x_{11}x_{16} - 6x_{11}x_{17} + 6x_{11}x_{18} - 3x_{11}x_{19} - 6x_{11}x_{20} + 2x_{12}x_{13} + x_{12}x_{14} - 3x_{12}x_{15} + 3x_{12}x_{16} + 2x_{12}x_{17} - 2x_{12}x_{18} + x_{12}x_{19} + 2x_{12}x_{20} - 2x_{13}x_{14} + 6x_{13}x_{15} - 6x_{13}x_{16} - 4x_{13}x_{17} + 4x_{13}x_{18} - 2x_{13}x_{19} - 4x_{13}x_{20} + 3x_{14}x_{15} - 3x_{14}x_{16} - 2x_{14}x_{17} + 2x_{14}x_{18} - x_{14}x_{19} - 2x_{14}x_{20} + 9x_{15}x_{16} + 6x_{15}x_{17} - 6x_{15}x_{18} + 3x_{15}x_{19} + 6x_{15}x_{20} - 6x_{16}x_{17} + 6x_{16}x_{18} - 3x_{16}x_{19} - 6x_{16}x_{20} + 4x_{17}x_{18} - 2x_{17}x_{19} - 4x_{17}x_{20} + 2x_{18}x_{19} + 4x_{18}x_{20} - 2x_{19}x_{20}$

subject to $26x_1 - 42x_2 + 11x_3 + 5x_4 + 33x_5 + 35x_6 + 42x_7 + 3x_8 - 49x_9 + 25x_{10} + 11x_{11} - 44x_{12} + 34x_{13} + 32x_{14} - 45x_{15} + 3x_{16} + 11x_{17} + 5x_{18} + 6x_{19} - 44x_{20} \leq 193$; $x_1 + 39x_2 + 25x_3 + 23x_4 + 25x_5 - 42x_6 - 49x_7 + 22x_8 + 3x_9 - 45x_{10} - 42x_{11} + 37x_{12} + 28x_{13} + 22x_{14} + 13x_{15} - 43x_{16} + 34x_{17} - 48x_{18} + 11x_{19} \leq 115$; $0 \leq x_i \leq 10$, integer.

Solution: $x_5 = 6,\ x_7 = 5,\ x_{12} = 2,\ x_{15} = 3,\ x_{16} = x_{18} = 1;\ x_i = 0$ otherwise.

Problem 6.

Maximize $90x_1 + 13x_2 + 56x_3 - 3x_1^2 + 7x_1x_4 + 14x_1x_5 + 19x_1x_6 - 2.5x_2^2 + 7x_2x_7 + 14x_2x_8 + 19x_2x_9 - 2.5x_3^2 + 7x_3x_{10} + 14x_3x_{11} + 19x_3x_{12} - 24.5x_4^2 - 98x_4x_5 - 133x_4x_6 - 98x_5^2 - 266x_5x_6 - 180.5x_6^2 - 24.5x_7^2 - 98x_7x_8 - 133x_7x_9 - 98x_8^2 - 266x_8x_9 - 180.5x_9^2 - 24.5x_{10}^2 - 98x_{10}x_{11} - 133x_{10}x_{12} - 98x_{11}^2 - 266x_{11}x_{12} - 180.5x_{12}^2$

subject to $x_4 + x_5 + x_6 = 1;\ x_7 + x_8 + x_9 = 1;\ x_{10} + x_{11} + x_{12} = 1;\ x_4 + x_7 + x_{10} = 1;\ x_5 + x_8 + x_{11} = 1;\ x_6 + x_9 + x_{12} = 1;\ x_1, x_2, x_3 \geq 0,$ $x_4, \cdots, x_{12} = 0$ or 1.

Solution: $x_1 = 18.17,\ x_2 = 4,\ x_3 = 14,\ x_6 = x_7 = x_{11} = 1;\ x_i = 0$ otherwise.

Appendix B. A Computational Scheme for Identifying Various Quadratic Concavities[2]

Step 1 Compute the eigenvalues λ_i ($i = 1, \cdots, n$) of the matrix D. If $\lambda_i \leq 0$ for all i, the function $\phi(x)$ is concave. Go to step 6(b). If only one eigenvalue is positive, go to step 2. Otherwise, $\phi(x)$ is nonquasiconcave Go to step 6(c)

Step 2 Check for the sign of the vector p If $p = 0$, go to step 3(a) If $p_i < 0$ for some i, go to step 6(c). If $p \geq 0$, go to step 3(b)

Step 3. Check for the sign of D

(a) If $D \geq 0$, $\phi(x)$ is quasiconcave Go to step 5 If $D < 0$ for some i, j, go to step 6(c)

(b) If $D \geq 0$, go to step 4 If $D < 0$ for some i, j, go to step 6(c).

Step 4. Calculate the eigenvalues of $\begin{bmatrix} D & p \\ p' & 0 \end{bmatrix}$. If only one eigenvalue is positive, $\phi(x)$ is pseudoconcave. Go to step 6(a). Otherwise, go to step 6(c).

Step 5. Replace some elements of p by arbitrarily small positive numbers to yield pseudoconcavity. Go to step 6(a).

Step 6. (a) Solve the discrete quadratic programming problem, using the functional value at central node as bound.

(b) Solve the discrete quadratic programming problems, estimating the change in functional value at branch nodes by the fall of the gradient function at central node

(c) Only the relaxed problem will be solved

When the objective function has been identified as nonquasiconcave on the nonnegative orthant, we could try to find a subset of E_+^n on which it may be quasiconcave (for instance when the saddle point of the function is not at the origin). This could be done by locating the saddle point of the function using ordinary calculus, and would in effect require the solution of a system of homogeneous linear equations. With respect to the new axes thus found, we then go back to step 2 to continue the test for quasiconcavity.

ACKNOWLEDGMENTS. I am grateful to Dr. A.H. Land for her guidance and encouragement throughout this work. I also wish to thank the referee for helpful suggestions.

REFERENCES

1. AGRAWAL, S.C. On mixed integer quadratic programs. *Naval Res. Logistics Quart 21* (1974), 289–297.
2. AGRAWAL, S C On integer solutions to quadratic programs by a branch and bound technique. *Trabajus de Estadistica y de Investigation Operational 25* (1974), 65–70.
3. BALAS, E. Duality in discrete programming. The quadratic case. *Manage. Sci. 16*, 1 (1969), 14–32.
4. COTTLE, R W., AND FERLAND, J A. On pseudo-convex functions of non-negative variables *Math. Programming 1*, 1 (1971), 95–101
5. COTTLE, R.W., AND FERLAND, J A. Matrix theoretic criteria for the quasi-convexity and pseudo-convexity of quadratic functions *Linear Algebra and Its Applications 5*, 2 (1972), 123–136
6. GREENBERG, H.J , AND PIÈRSKALLA, W.P. A review of quasi-convex functions *Oper. Res 19*, 7 (1971), 1553–1570.

[2] The scheme is based on various results reported in [4, 5, 14].

7. KUNZI, H.P., AND OETTLI, W. Integer quadratic programming In *Recent Advances in Mathematical Programming*, Robert L Graves and Philip Wolfe, Eds., McGraw-Hill, New York, 1963, pp. 303–308
8 LAND, A.H., AND DOIG, A G An automatic method of solving discrete programming problems. *Econometrica 28*, 3 (1960), 497–520
9. LAND, A H , AND MORTON, G An inverse-basis method for Beale's quadratic programming algorithm *Manage. Sci 19*, 5 (1973), 510–516
10 LAND, A H , AND POWELL, S Fortran codes for mathematical programming. Linear, quadratic and discrete. Wiley, New York, 1973
11. LAUGHHUNN, D J. Quadratic binary programming with application to capital budgeting problems *Oper Res 18*, 3 (1970), 454–461
12. MANGASARIAN, O L *Nonlinear Programming*. McGraw-Hill, New York, 1969
13 MAO, J T C , AND WALLINGFORD, B A An Extension of Lawler and Bell's method of discrete optimisation with examples from capital budgeting *Manage. Sci 15*, 2 (1968), B51–B60
14 MARTOS, B Quadratic programming with a quasiconvex objective function *Oper. Res 19*, 1 (1971), 87–97
15 MYLANDER, W C Finite algorithms for solving quasiconvex quadratic programs *Oper. Res 20*, 1 (1972), 167–173
16. PHUONG, T.H An algorithm for solving quadratic programs having integer variables Available c/o Dr A.H. Land, London School of Economics, Houghton Street, London WC2, England.
17 PONSTEIN, J. Seven kinds of convexity *SIAM Rev 9*, 1 (1967), 115–119
18. ROSEN, J B , AND SUZUKI, S. Construction of non-linear programming test problems *Comm. ACM 8*, 2 (Feb 1965), 113

RECEIVED JANUARY 1975; REVISED OCTOBER 1975