MANAGEMENT SCIENCE
Vol. 12, No. 11, July, 1966
Printed in U.S.A.

DISCRETE OPTIMIZING SOLUTION PROCEDURES FOR LINEAR AND NONLINEAR INTEGER PROGRAMMING PROBLEMS*†

STANLEY REITER AND DONALD B. RICE

Purdue University

We present a method for approximating the solution of mixed integer nonconcave programming problems in bounded variables. We present computational results for 39 test problems which suggest that the procedure offers a practical useful way of approximating solutions of programming problems of the type tested. We have also discovered an apparent regularity in the distribution of local optima generated by the method; the method seems to generate Beta distributions in all problems for which results are available. This suggests interesting opportunities for further exploration.

Introduction

In this paper we present a procedure for approximating the solution of mixed integer nonconcave programming problems in bounded variables. This procedure is an application of the discrete optimizing approach presented in [2]. We have programmed this procedure for the less general class of problems in which all functions are nonhomogeneous quadratic forms, including the (degerate) case of linear forms, and applied these programs to 39 test problems ranging in size from 10 to 90 variables and 2 to 15 constraints.[1] We have also worked with some nonconcave problems with good success but we do not report these results here because our experience is still too limited.

The results we have gotten with our 39 test problems suggest very strongly that this solution procedure is a practical, useful way of closely approximating the solutions to programming problems of the type tested. The significance of being able to solve problems like these will be clear to anyone from a brief examination of, say, Dantzig's chapter on integer programming problems [1]. This chapter concerns itself only with problems that can be formulated as all-linear integer programming problems. But we may expect that a rich variety of problems can be given nonlinear mixed integer formulations.

We should have liked to compare the performance of this procedure with that of other computational methods on the same problems. However, we have not been able to do so.

It is likely that there is no uniformly best method for problems of this type. Some methods will no doubt work better in some cases than others. We feel that experience with a wide variety of problems will be necessary before we can decide which methods are best for which classes of problems.

* Received June 1965 and revised March 1966.

† This research was supported by Office of Naval Research Contract Nonr-1100(16).

[1] The bounds on the variables (equation (3) in the next section) contribute another 20 to 180 constraints. These constraints are not included in the figures reported here which include only functional constraints.

We have not found any examples of real problems formulated as nonconcave mixed integer problems, although we have heard that some exist. Perhaps the availability of a method of solution will stimulate the formulation of such problems.

Description of the Problem

This paper describes a computing procedure for integer programming problems, specifically for the following types of problems:

$$\text{maximize } G'X + \tfrac{1}{2}X'CX \tag{1}$$

subject to

$$D'_iX - \tfrac{1}{2}X'H_iX \leqq E_i, \qquad i = 1, \cdots, m \tag{2}$$

$$L \leqq X \leqq U \tag{3}$$

$$X \text{ integral} \tag{4}$$

where X is an n-vector, X' its transpose, G, L, U, and D_i, $i = 1, \cdots, m$, are n-vectors, C and H_i, $i = 1, \cdots, m$ are negative semidefinite matrices, and E_i, $i = 1, \cdots, m$ is a scalar.

We distinguish three subclases of problems:

1. LL, where $C = H_i \equiv 0$, $\quad i = 1, \cdots, m$
2. QL, where $H_i \equiv 0$, $\quad i = 1, \cdots, m$
3. QQ, where $C \neq 0$, $H_i \neq 0$, $\quad i = 1, \cdots, m$.

The Solution Procedure

The solution procedure consists of (a) choosing a random starting point, (b) locating a feasible point, (c) applying a modified gradient maximizing procedure. This procedure determines a probability distribution on the set of values taken by the objective function at feasible lattice points. This distribution gives nonzero probability to values of the objective function which are local maxima relative to the search procedure. The estimated solution to the programming problem is the maximum value (and the corresponding lattice point) observed in repeated samples from this distribution. Thus this method is a particular instance of the discrete optimizing technique [2].

For step (a) we assume that we have independent upper and lower limits L and U on X. Equation (3), together with (4), defines a finite set of points enclosed by a hyperrectangle in n-space. We sample with a uniform distribution from the set of integers in the interval $[L_j, U_j]$, $j = 1, \cdots, n$. This results in uniform sampling of lattice points in the hyperrectangle.

If the point sampled from the hyperrectangle is feasible (satisfies (2)) we go directly to step (c) described below. If the point X^+ sampled is not feasible we perform a search by the "method of weighted perpendiculars" (WP) as follows. The quantity S_i

$$S_i = D'_iX^+ - \tfrac{1}{2}X^{+\prime}H_iX^+ - E_i, \qquad i = 1, \cdots, m \tag{5}$$

is the amount by which the i^{th} constraint is violated at X^+. We define

$$(6)\qquad Z = -\Big(\textstyle\sum_{i:\,S_i>0} S_i[\nabla_i(D'_iX - \frac{1}{2}X'H_iX)]_{X=X^+}\Big)\Big/\textstyle\sum_{i:\,S_i>0} S_i$$

where ∇_i denotes the vector of partial derivatives of the i^{th} constraint with respect to the x_j's. The vector $-Z$ is thus a positive linear combination of these perpendiculars and therefore lies in the convex cone spanned by them. Let

$$(7)\qquad Z_j^{\#} = \min_{Z_j \neq 0} |Z_j|$$

Define $Z^+ = (Z_1^+, \cdots, Z_n^+)$

$$(8)\qquad Z_j^+ = \begin{cases} [(Z_j/Z_j^{\#}) + 0.5], & \text{if} \quad Z_j \geqq 0 \\ [(Z_j/Z_j^{\#}) - 0.5], & \text{if} \quad Z_j \leqq 0 \end{cases} \qquad j = 1, \cdots, n$$

where $[w]$ denotes the largest integer not exceeding w. Note that Z^+ has the following properties:

$$(9)\qquad \min_{Z_j^+ \neq 0} |Z_j^+| = 1$$

$$(10)\qquad Z_j^+ = 0 \quad \text{if and only if} \quad Z_j = 0$$

$$(11)\qquad Z_j^+ \quad \text{is an integer,} \quad j = 1, \cdots, n.$$

The equation

$$(12)\qquad Y_b = X^+ + b \cdot Z^+$$

is the equation of a line passing through the point X^+. This line may or may not have integer points on it. Taking successively the values $b = 1, 2, 3, \cdots$, we test the corresponding Y_b, necessarily a lattice point, for feasibility; we eventually either locate a feasible point or violate one or more of the upper or lower limits. If one or more of these upper or lower limits is violated we hold the corresponding coordinates of Y_b constant at those limits and continue to search by increasing b. Eventually we find a feasible point or all coordinates go outside of their limits. In the second case we abandon X^+ and sample another starting point from the hyperrectangle.

Starting with a nonfeasible point we look for a feasible lattice point by searching along a line chosen so as to be likely to intersect the feasible region and along which lattice points are characterized by integer values of the path parameter. See Figure 1. There are some cases in which the feasible region is confined to a small portion of the hyperrectangle, tucked away in a remote corner so to speak. In some of these cases the WP method may require several trials to find a feasible point. In such cases another method called "permuted coordinates" (PC) may be better. This method is applicable when $L = 0$, or when all constraints are linear.

The method is as follows. Generate a permutation I of the integers from 1 to n,

$$(13)\qquad I = (i_1, i_2, \cdots, i_n).$$

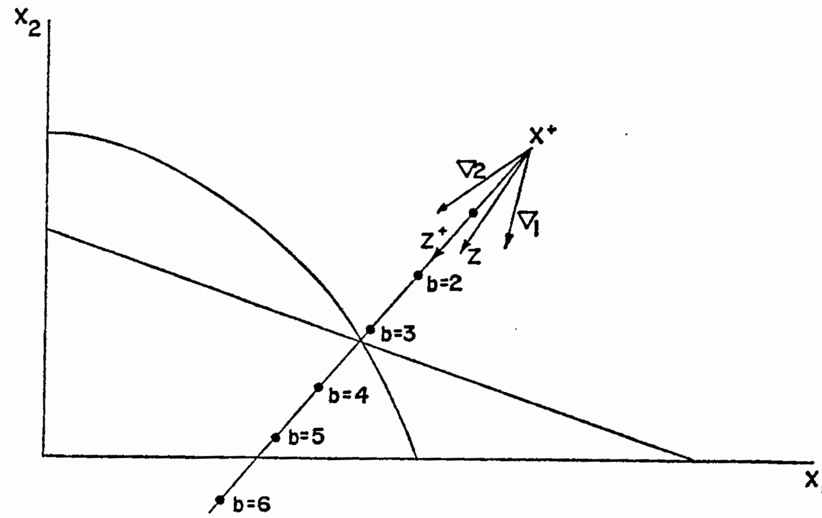

FIG. 1. Representation of *WP* method

Sample an integer value x_{i_1} from $[L_{i_1}, U_{i_1}]$. Define $T_1 = (0, \cdots, 0, x_{i_1}, 0, \cdots, 0)$ and $Z_2 = (0, \cdots, 0, 1, 0, \cdots, 0)$ where the one is in the i_2^{th} place. Construct

$$T_2 = T_1 + b_2 \cdot Z_2 . \tag{14}$$

Note however that T_2 must satisfy

$$D'_i T_2 - \tfrac{1}{2} T'_2 H_i T_2 \leqq E_i , \qquad i = 1, \cdots, m \tag{15}$$

$$L \leqq T_2 \leqq U. \tag{16}$$

Recall that H_i is negative semidefinite. Statements (14), (15), and (16) imply

$$L_{b_2} \leqq b_2 \leqq U_{b_2} \tag{17}$$

where $L_{b_2} \geqq L_{i_2}$ and $U_{b_2} \leqq U_{i_2}$. If (17) does not contain an integer we start over with a new x_{i_1}. If the interval in (17) contains an integer we select an integer value by uniform sampling. With this value we calculate T_2 and then,

$$T_3 = T_2 + b_3 \cdot Z_3 \tag{18}$$

where Z_3 is a vector with the value one in coordinate i_3 and zero in all other coordinates.

We continue until a value b_n is sampled and T_n is calculated. T_n is a feasible lattice point.

Having located a feasible lattice point T^+ we start step (c). In step (c) we end up with a locally maximal lattice point.

Let $R(X)$ denote the gradient of the objective function, i.e.,

$$R(X) = \nabla(G'X + \tfrac{1}{2} X'CX). \tag{19}$$

Define

$$Y_d = T^+ + d \cdot R(T^+). \tag{20}$$

Notice that

$$R_j(T^+) = G_j + C_j T^+, \tag{21}$$

where C_j is the j^{th} row of C. Since R is not in general integer valued we apply the procedure used in the WP method to obtain a normalized integer coordinate vector R^+. Equations (7) through (11) give the desired result, with R_j substituted for Z_j, R^+ for Z^+ and $R_j^{\#}$ for $Z_j^{\#}$. The vector R^+ is the modified gradient of the objective function and

$$Y_d = T^+ + d \cdot R^+ \tag{22}$$

defines the path on which we search for improved values of the objective function.

We find that value of d for which $f(Y_d)$ is a maximum over feasible Y_d on this path. This may be done in one of two ways: (1) Let $d = 0, 1, 2, 3, \cdots$, successively and find the first value of d for which either Y_{d+1} is infeasible or $f(Y_{d+1}) \leqq f(Y_d)$. (2) Solve directly for the maximizing value of d. Evaluating f at Y_d we have

$$f(d) = \tfrac{1}{2}(R^{+\prime}CR^+)\, d^2 + (G'R^+ + R^+CT^+)\, d + (G' + \tfrac{1}{2}T^{+\prime}C)T^+. \tag{23}$$

For C negative definite $f(d)$ is concave in d with absolute maximum at

$$d^0 = -(G'R^+ + R^{+\prime}CT^+)/R^{+\prime}CR^+. \tag{24}$$

The unconstrained optimal integer value of d is

$$d^{\#} = \max_{d_1, d_2} \{f(T^+ + d_1R^+), f(T^+ + d_2R^+)\}, \tag{25}$$

where

$$d_1 = [d^0] \tag{26}$$

and

$$d_2 = [d^0 + 1]. \tag{27}$$

Let U_d and L_d be the largest and smallest values of d for which Y_d satisfies (2) and (3). If $d^{\#} > U_d$, then the best value of d is given by $[U_d]$. If $d^{\#} < L_d$, then the best value of d is given by $[L_d + 1]$. If $L_d \leqq d^{\#} \leqq U_d$, then $d^{\#}$ is the best value of d. Note that $L_d \leqq 0 \leqq U_d$. If the objective function is linear the best value of d is $[U_d]$ and the calculation is much simpler.

If the best value of d, denoted d^*, is not zero, we set $T^+ = Y_{d^*}$ and repeat the procedure. If $d^* = 0$ we set the element of $R(T^+)$ of smallest nonzero absolute value equal to zero to obtain $R^1(T^+)$ and proceed as before with R^1 in place of R. If R^1 leads to no improvement, set the element of R^1 of smallest nonzero absolute value equal to zero to obtain $R^2(T^+)$ and proceed with R^2 in place of R. This procedure continues until R^v is obtained with only one nonzero element. If using $R^v(T^+)$ results in no improvement we try successively R^{v+1} to R^{v+k}, where

$k + 1$ is the number of zonzero elements of R and R^{v+j} is a vector with 1 in the coordinate corresponding to the partial derivative in R which ranks $(j + 1)$ in absolute value, for $j \leqq k$. Ties are broken in order of the subscript of the corresponding variable.

If R^{v+k} results in no improvement then T^+ is a locally maximal point. If any R^j leads to improvement, we set T^+ equal to the Y_{d*} thus obtained and repeat the procedure.

Learning

Let $T_1^0, \cdots, T_r^0$ be the first r locally maximal points and let

$$(28) \qquad T^0 = ((T_{ij}^0)) \quad i = 1, \cdots, r; \quad j = 1, \cdots, n$$

be the matrix whose rows are r locally optimal n-vectors. Define

$$(29) \qquad L_j^\# = \min_i T_{ij}^0, \qquad j = 1, \cdots, n$$

and

$$(30) \qquad U_j^\# = \max_i T_{ij}^0, \qquad j = 1, \cdots, n.$$

Then

$$(31) \qquad P^\# = \{X : L^\# \leqq X \leqq U^\#\}$$

defines a hyperrectangle which contains these r points. The region $P^\#$ is, we hope, a better place to look for large values of the objective function than the original hyperrectangle. However, since we can't be sure that optimal points are to be found only there we sample starting points from that region only a fraction of the time.

At the beginning of the k^{th} iteration, $k > r$, we conduct a two outcome experiment with probability of success equal to q. If a success occurs the k^{th} starting point is sampled from $P^\#$; if a failure occurs, the k^{th} starting point is sampled from the original hyperrectangle. Whenever a local maximum is found which is among the best r points found so far, a new hyperrectangle is calculated using the new matrix T^0 formed by the new set of r best points. Using this procedure we sample with probability q from the smallest hyperrectangle "parallel" to the axes which encloses the best r points found so far. Thus the starting point sampling process has to parameters, q and r, where q is the probability of sampling from the smaller region on iteration $k > r$, and r is the number of locally maximal points saved.

Extensions to Mixed Integer or Nonconcave Problems

The solution procedure just described can be modified to apply to mixed integer problems. Other minor modifications can make these methods applicable to problems in which the objective function and constraints are not quadratic or indeed not even concave. Thus, for example, these methods can be applied to multimodal functions with only some variables required to be integer.

To handle a mixed problem with $n' < n$ variables required to be integer valued we renumber the variables so that the first n' are the integer valued ones. Equation (7) and (8) for Z^+ and R^+ are replaced by

$$Z_j^{\#} = \min \{ | Z_j | : Z_j \neq 0, 1 \leqq j \leqq n' \} \tag{32}$$

$$Z_j^+ = \begin{cases} [(Z_j/Z_j^{\#}) + 0.5], & \text{if} \quad Z_j \geqq 0 \quad \text{and} \quad 1 \leqq j \leqq n' \\ [(Z_j/Z_j^{\#}) - 0.5], & \text{if} \quad Z_j \leqq 0 \quad \text{and} \quad 1 \leqq j \leqq n' \\ (Z_j/Z_j^{\#}), & \text{if} \quad n' < j \leqq n \end{cases} \tag{33}$$

As long as the gradient $R^j(T^+)$ has some nonzero values among its first n' coordinates the procedures described earlier, which restrict attention to integer values of the path parameter, apply directly. If all of the first n' coordinates of $R^j(T^+)$ are zero, it is unnecessary to normalize and adjust to integers. But it is necessary to solve directly for the best (noninteger) value of the path parameter, using the method of the second alternative described above involving equations (23) through (27). In the case of a problem where the objective function is not quadratic and concave, a grid must be superimposed on the modified gradient path and the method of alternative (1) applied.

The WP method can be used for finding feasible points whenever the partial derivatives of the constraint equations can be calculated at any point in the hyperrectangle. Integer values are required for only the first n' coordinates of the starting point.

Our procedures depend on being able to calculate a modified gradient path from any feasible point, and a WP path from any nonfeasible starting point. If the objective and constraints are differentiable functions, these paths can be calculated by means of the gradient. Even in the case of functions and constraints which are not differentiable, paths of positive ascent calculated by means of first differences can be used in place of the gradient. In this case the objective function and constraints need be defined only at lattice points.

If the objective function is a multimodal function the gradient maximizing procedure will of course locate local optima near each mode. This offers no particular difficulty since the method produces a distribution of local optima in any case.

We should note that the method is not well suited to dealing with equality constraints.

Test Problems

We have applied the solution procedures described above to 39 test problems, 13 each of the QQ, QL, and LL type. For 36 of the 39 problems we used a complete factorial design in each problem class. In the case of QQ problems the values of n were 10, 20, 30 and 40 and of m were 2, 5 and 10. In the QL case values of n were 10, 20, 30 and 50 and of m were 5, 10 and 15. In the LL case values of n were 10, 30, 60 and 90 with values of 5, 10 and 15 for m. These 36 problems were constructed to have noninteger solutions when the integer restrictions are disregarded, so that the integer solution differed from the solution to the corresponding concave programming problem. The remaining three problems consisted of one 10 variable two constraint problem of each type which were given integer solutions to begin with.

We constructed these problems as follows. The matrices C and H_i, $i = 1, \cdots, m$, of the quadratic forms were obtained from matrices of correlation coefficients which we took from a large multiple regression problem using actual data. We got the elements of the matrix D by random sampling with a uniform distribution on the integers zero to nine. The vector L was the zero vector in all 39 problems. Having specified C, D and H_i, $i = 1, \cdots, m$, we specified a desired optimal point X^* and a vector of corresponding Lagrangian variables Y^*. We used the procedure of Rosen and Suzuki [3] to calculate the values for G and E which result in a concave programming problem having the desired solution with the given constraints and objective function.

Specifically, given C, D, H_i, $i = 1, \cdots, m$, and the desired solution (X^*, Y^*) the Kuhn-Tucker theorem tells us that (X^*, Y^*) is the solution of

$$\max \{Q(X, Y) = G'X + \tfrac{1}{2}X'CX - \textstyle\sum_{i=1}^{m} Y_i(D'_iX - \tfrac{1}{2}X'H_iX - E_i)\}. \tag{34}$$

Then for C and H_i, $i = 1, \cdots, m$, negative definite E may be calculated from

$$E_i = \begin{cases} D'_iX^* - \tfrac{1}{2}X^{*\prime}H_iX^*, & \text{if } Y_i^* > 0 \\ D'_iX^* - \tfrac{1}{2}X^{*\prime}H_iX^* + w, & \text{if } Y_i^* = 0 \end{cases} \tag{35}$$

where w is any positive constant. Similarly

$$G = -CX^* + \textstyle\sum_{i=1}^{m} Y_i^*(D_i - H_iX^*). \tag{36}$$

That satisfies the Kuhn-Tucker condition

$$\nabla Q(X, Y)]_{X^*, Y^*} = 0 \tag{37}$$

which guarantees that X^* is an optimum solution to this concave programming problem.

Results—General Discussion

The method of solution described above leads, as we have seen, to random sampling from a distribution of local maxima determined by the problem, the local search procedure and the starting point sampling rules used. Therefore our test results are conveniently presented as observed frequency distributions of local maxima, and characteristics of the solution procedure are well described in terms of characteristics of these frequency distributions. Since we have used both learning, with parameters q and r, and nonlearning, we have frequency distributions corresponding to each of these types of search. The range of function values, the size of the feasible region, and the value of the unrestricted optimum $f(X^*)$ vary greatly from problem to problem. Therefore we normalized the observed frequency distribution by the transformation $f(X)/f(X^*)$ in order to facilitate comparisons of results among different problems. The normalized frequency distributions are given in Tables 1 through 12.

We are of course primarily interested in the upper tails of these distributions, since the best value observed is our estimate of the integer solution. To facilitate the description of the upper tail we distinguish two intervals. Consider the value obtained by truncating X^*, the unrestricted solution, to the vector whose coordinates are $[x_j^*]$. The corresponding normalized value gives us the lower end point of an interval whose upper end point is one. We are interested in the relative frequency in this interval, in the time required to obtain an observation in this interval, and in the relative frequency in the upper half of that interval and the time required to obtain an observation there. The percentage error in the observed solution to these problems is bounded by 1 minus the normalized best value found.

We give first the following dictionary of symbols appearing in the tables:

f = normalized objective function values
f_T = value of f for the truncated unrestricted solution
f^0 = best function value found
t_0 = computer time per observation in minutes
n = number of variables
m = number of functional constraints
N = number of observations taken
$\%_I = (f^0 - f_T)/(f^* - f_T)$
r_I = relative frequency of $[f_T, f^*]$
r_H = relative frequency of $[(f_T + f^*)/2, f^*]$
t_I = computer time to reach f_T in minutes
t_H = computer time to reach $(f_T + f^*)/2$ in minutes
f' = $10f - 9$

Let the pair (p, q) be parameters of the beta density function of form

$$g(z; p, q) = (1/B(p, q))z^{p-1}(1 - z)^{q-1}$$

and let K be a value of the Kolmogoroff-Smirnov goodness of fit test statistic. The symbols p, q, and K are discussed subsequently.

Interpreting the Tables of Results

Table 1 to 12 summarize the results from the test problems, numbered 1 to 36, for the ladder learning and nonlearning procedures. Tables 1 to 4 describe the QQ output, 5 to 8 the QL output, and 9 to 12 the LL output.

Consider Table 1 which summarizes problems 1 to 12 in the QQ class. Problem 1 has 40 variables and 10 functional constraints. The function value of the truncated unrestricted solution is 0.9717. This is the value that would be obtained if the unrestricted solution were known and rounded down to obtain an integer solution to the problem. Computer time per observation was 1.506 minutes.

When 35 ladder learning observations were taken in a single continuous run on problem 1 the best function value found among these 35 local optima was 0.9847. This value corresponds to 0.4598 of the interval from f_T to $f^* = 1$. The relative frequency of this interval was 0.686 and of the upper half of this interval

TABLE 1

Summary of Output and Problem Data, QQ Class, Problems 1–12, Learning and Non-Learning Procedures

Problem Data					Learning Output							Non-learning Output						
No.	n	m	f_T	t_0	N	f^0	$\%_I$	r_I	r_H	t_I	t_H	N	f^0	$\%_I$	r_I	r_H	t_I	t_H
1	40	10	.9717	1.51	35	.9847	.4598	.69	.00	12.0	—	73	.9850	.4699	.25	.00	7.53	—
2	40	5	.9739	1.69	34	.9898	.6073	.97	.18	13.5	16.9	75	.9886	.5611	.35	.01	5.08	128.
3	40	2	.9792	.767	36	.9901	.5235	.81	.06	6.90	31.4	71	.9873	.3919	.27	.00	3.07	—
4	30	10	.9724	.721	33	.9815	.3300	.45	.00	8.65	—	81	.9770	.1652	.02	.00	29.6	—
5	30	5	.9742	.618	36	.9938	.7611	.97	.58	5.56	6.80	79	.9850	.4188	.28	.00	2.47	—
6	30	2	.9795	.266	37	.9956	.7863	.97	.89	2.40	2.93	77	.9886	.4413	.26	.00	1.07	—
7	20	10	.9722	.217	75	.9842	.4302	.29	.00	6.95	—	87	.9675	—	.00	.00	—	—
8	20	5	.9748	.185	35	.9932	.7283	.37	.06	1.49	2.40	680	.9851	.4082	.06	.00	3.14	—
9	20	2	.9796	.098	34	.9925	.6310	1.0	.12	0.79	1.77	76	.9874	.3813	.17	.00	0.59	—
10	10	10	.9719	.036	34	.9940	.7885	.76	.65	0.36	0.58	74	.9825	.3781	.07	.00	0.54	—
11	10	5	.9745	.029	34	.9967	.8714	1.0	.73	0.23	0.29	78	.9932	.7322	.26	.04	0.11	0.7[illegible]
12	10	2	.9782	.018	35	.9947	.7570	1.0	.57	0.14	0.23	76	.9924	.6509	.39	.07	0.05	0.2[illegible]

TABLE 2

Summary of Output and Problem Data for Repeated Runs on Problem 8, QQ Class, Learning

Problem Data					Learning Output						
No.	n	m	f_T	t_0	N	f^0	$\%_I$	r_I	r_H	t_I	t_H
8	20	5	.9748	.133	147	.9913	.6547	.84	.09	1.81	12.4
8	20	5	.9748	.140	154	.9859	.4396	.45	.00	5.48	—
8	20	5	.9748	.148	147	.9905	.6244	.79	.15	1.63	1.63
8	20	5	.9748	.146	140	.9954	.8186	.89	.56	2.49	4.54
8	20	5	.9748	.133	145	.9944	.7790	.90	.43	1.99	8.75
8	20	5	.9748	.156	65	.9925	.7025	.75	.29	1.25	2.65
8	20	5	.9748	.157	69	.9929	.7187	.90	.26	1.26	4.09
8	20	5	.9748	.149	68	.9923	.6929	.75	.16	1.19	7.89
8	20	5	.9748	.130	69	.9860	.4451	.81	.00	3.12	—
8	20	5	.9748	.138	68	.9857	.4338	.37	.00	5.66	—
8	20	5	.9748	.164	38	.9797	.1926	.03	.00	8.06	—
8	20	5	.9748	.176	40	.9875	.5027	.57	.02	1.76	7.94
8	20	5	.9748	.154	31	.9822	.2931	.23	.00	4.30	—
8	20	5	.9748	.152	31	.9858	.4375	.35	.00	1.22	—
8	20	5	.9748	.172	28	.9844	.3794	.46	.00	2.57	—
8	20	5	.9748	.175	27	.9909	.6377	.48	.11	1.40	2.98
8	20	5	.9748	.176	35	.9929	.7187	.80	.14	1.41	4.59
8	20	5	.9748	.167	32	.9862	.4516	.47	.00	1.33	—
8	20	5	.9748	.146	33	.9841	.3700	.61	.00	3.51	—
8	20	5	.9748	.155	30	.9851	.4082	.20	.00	6.35	—

was zero. Actual computer time elapsed before finding a value greater than or equal to f_T was 12.05 minutes. Since no value greater than the midpoint of the interval was located, no observation is available on t_H for problem 1.

On the same problem a composite sample of 73 independent nonlearning observations produced a best function value of 0.9850 which corresponds to 0.4699

TABLE 3

Distributions of Normalized Function Values for QQ Class Problems 1–12 with Learning

Range	Prob. No. 1	2	3	4	5	6	7	8	9	10	11	12
	n											
	40			30			20			10		
	m											
	10	5	2	10	5	2	10	5	2	10	5	2
.995–1.00	—	—	—	—	—	.05	—	—	—	—	.06	—
.990–.995	—	—	.03	—	.33	.81	—	.06	.12	.32	.47	.46
.985–.990	—	.38	.47	—	.36	.08	—	.03	.56	.35	.26	.43
.980–.985	.23	.41	.28	.06	.19	.03	.07	.06	.29	—	.12	.11
.975–.980	.26	.15	.11	.30	.08	—	.12	.20	.03	.03	.09	—
.970–.975	.34	.03	—	.30	—	—	.17	.14	—	.06	—	—
.965–.970	.06	—	.03	.12	.03	—	.20	.17	—	.06	—	—
.960–.965	—	—	—	.09	—	—	.12	.20	—	.03	—	—
.955–.960	—	—	—	.09	—	—	.07	.09	—	.06	—	—
.950–.955	—	—	—	.03	—	—	.13	.03	—	.03	—	—
.945–.950	—	—	—	—	—	—	.07	.03	—	.03	—	—
.940–.945	—	—	—	—	—	—	.03	—	—	—	—	—
.935–.940	—	—	—	—	—	—	.03	—	—	.03	—	—
.930–.935	—	—	—	—	—	—	—	—	—	—	—	—
.925–.930	—	—	—	—	—	—	—	—	—	—	—	—
.920–.925	—	—	—	—	—	—	—	—	—	—	—	—
.915–.920	—	—	—	—	—	—	—	—	—	—	—	—
.910–.915	—	—	—	—	—	—	—	—	—	—	—	—
.905–.910	.03	—	—	—	—	—	—	—	—	—	—	—
.900–.905	—	—	—	—	—	—	—	—	—	—	—	—
Below .90	.08	.03	.08	—	—	.03	—	—	—	—	—	—
N	35	34	36	33	36	37	75	35	34	34	34	35
*P	40	53	50	33	34	57	10	†	49	10	46	52
*q	13	11	9.5	13	5	5	5.5	†	8	2	5.5	6.5
K	.04	.06	.06	.04	.04	.10	.03	†	.08	.10	.03	.04

* Beta parameters are for transformed variable, f'.

† Not available, see Table 4.

of the interval $[f_T, f^*]$. Relative frequencies of this interval and its upper half were 0.247 and 0.000, respectively. Expected time to reach the interval based on the relative frequencies of this composite sample was 7.53 minutes. Again with no observed values in the upper half of the interval there is no estimate available for time required to reach it.

Nonlearning observations from several different runs on the same problem can be combined to form a pooled sample, since they are independent observations,

TABLE 4

Repeated Observations on Distributions of Normalized Function Values for Problem No. 8, QQ Class with Learning

Range	Problem No. 8, QQ, $n = 20$, $m = 5$																			
.995–1.00	—	—	—	.04	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.990–.995	.01	—	.03	.44	.14	.14	.10	.09	—	—	—	—	—	—	—	.04	.06	—	—	—
.985–.990	.25	.03	.29	.19	.46	.26	.32	.21	.12	.06	—	.13	—	.03	—	.07	.26	.06	—	.03
.980–.985	.33	.16	.28	.15	.23	.25	.35	.34	.61	.13	—	.25	.10	.13	.21	.22	.31	.22	.49	.03
.975–.980	.25	.26	.19	.07	.06	.09	.13	.12	.09	.18	.03	.20	.13	.19	.25	.11	.17	.19	.12	.13
.970–.975	.10	.14	.06	.04	.04	.12	.07	.10	.04	.22	.03	.15	.13	.16	.11	.22	.14	.22	.09	.07
.965–.970	.03	.11	.08	.01	.01	.03	.01	.02	.03	.15	.03	.18	.13	.07	.14	.07	.03	.03	.06	.17
.960–.965	.01	.08	.04	.02	.01	.05	—	.01	.01	.07	.06	.07	.03	.10	.11	.11	—	.03	.03	.13
.955–.960	—	.04	—	.01	.01	.02	—	—	.04	—	.08	—	.13	—	.04	.04	—	—	.09	—
.950–.955	—	.04	.01	.01	—	.03	.01	—	.03	—	.11	—	.10	.07	.04	.07	.03	—	.06	—
.945–.950	.01	.05	.01	—	—	.01	—	.02	—	.02	.16	.02	.03	.03	—	.04	—	.03	—	.03
.940–.945	—	.01	—	—	.02	—	—	.04	—	.04	.18	—	.16	.03	—	—	—	.09	—	.10
.935–.940	—	.03	.01	—	—	—	—	—	—	.04	.10	—	.03	.10	—	—	—	—	—	.10
.930–.935	—	.01	—	—	—	—	—	—	—	.04	.08	—	.03	—	—	—	—	—	—	.10
.925–.930	.01	.01	—	—	—	—	—	—	.01	—	.05	—	—	.03	—	—	—	—	.03	—
.920–.925	.01	.02	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.915–.920	—	.01	—	.01	—	—	—	—	—	.02	—	—	—	—	—	—	—	—	—	.03
.910–.915	—	—	—	—	—	—	—	.01	—	.02	—	—	—	—	—	—	—	.03	—	.03
.905–.910	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.900–.905	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Below .90	—	—	.01	.01	.01	—	—	.04	.01	.02	—	—	—	.06	.11	—	—	.09	.03	.03
N	147	154	147	140	145	65	69	68	69	68	38	40	31	31	28	27	35	32	33	30
$*P$	15	33	33	33	32	12	28	17	47	8.5	†	22	6.5	†	22	11	21	†	†	†
$*q$	4	8	8	4.5	5.5	3	5.5	4	11	4	†	7	4	†	8	4	5	†	†	†
K	.08	.03	.03	.08	.08	.05	.05	.05	.10	.11	†	.05	.04	†	.05	.04	.04	†	†	†

* Beta parameters are for transformed variable, f'.
† Not available.

TABLE 5

Summary of Output and Problem Data, QL Class, Problems 13–24, Learning and Non-Learning Procedures

Problem Data					Learning Output							Non-Learning Output						
No.	n	m	f_T	t_0	N	f^0	$\%_I$	r_I	r_H	t_I	t_H	N	f^0	$\%_I$	r_I	r_H	t_I	t_H
13	50	15	.9810	1.55	34	.9934	.6544	.91	.09	15.5	46.7	150	.9909	.5215	.09	.01	17.1	118.
14	50	10	.9820	1.61	34	.9882	.3428	.53	.00	17.7	—	78	.9870	.2759	.05	.00	32.1	—
15	50	5	.9893	1.62	34	.9918	.2279	.38	.00	29.1	—	81	.9911	.1649	.02	.00	66.3	—
16	30	15	.9785	.295	36	.9896	.5140	.61	.03	2.66	14.8	73	.9829	.2047	.04	.00	7.38	—
17	30	10	.9797	.314	38	.9920	.6039	.74	.05	2.83	10.4	76	.9892	.4681	.05	.00	6.29	—
18	30	5	.9867	.284	38	.9909	.3150	.29	.00	3.41	—	71	.9926	.4403	.07	.00	4.27	—
19	20	15	.9772	.084	36	.9876	.4570	.81	.00	0.84	—	75	.9816	.1937	.07	.00	1.34	—
20	20	10	.9782	.077	31	.9884	.4666	.29	.00	1.46	—	668	.9896	.5212	.07	.00	1.38	57.8
21	20	5	.9854	.065	37	.9974	.8199	.97	.70	0.52	0.65	73	.9892	.2605	.10	.00	0.71	—
22	10	15	.9747	.017	33	.9897	.5916	.88	.21	0.15	0.50	96	.9846	.3906	.02	.00	0.82	—
23	10	10	.9756	.017	37	.9918	.6619	.84	.13	0.13	0.52	95	.9895	.5684	.07	.01	0.23	1.61
24	10	5	.9807	.016	36	.9978	.8839	.97	.81	0.14	0.14	93	.9957	.7794	.35	.06	0.05	0.25

TABLE 6

Summary of Output and Problem Data for Repeated Runs on Problem 20, QL Class, Learning

Problem Data					Learning Output						
No.	n	m	f_T	t_0	N	f^0	$\%_I$	r_I	r_H	t_I	t_H
20	20	10	.9782	.064	142	.9933	.6910	.96	.24	0.90	1.03
20	20	10	.9782	.068	153	.9897	.5271	.87	.01	1.09	10.9
20	20	10	.9782	.064	149	.9916	.6140	.91	.22	1.03	1.74
20	20	10	.9782	.071	150	.9942	.7354	.93	.23	0.99	1.20
20	20	10	.9782	.068	145	.9911	.5904	.92	.12	0.68	4.02
20	20	10	.9782	.086	62	.9901	.5451	.84	.03	1.29	3.60
20	20	10	.9782	.069	66	.9881	.4562	.82	.00	1.59	—
20	20	10	.9782	.077	74	.9874	.4236	.84	.00	1.08	—
20	20	10	.9782	.075	72	.9919	.6299	.86	.07	0.82	1.65
20	20	10	.9782	.074	69	.9884	.4698	.43	.00	1.40	—
20	20	10	.9782	.079	38	.9903	.5532	.84	.10	0.63	1.11
20	20	10	.9782	.085	31	.9890	.4962	.61	.00	2.13	—
20	20	10	.9782	.076	30	.9875	.4253	.57	.00	1.28	—
20	20	10	.9782	.084	32	.9888	.4864	.78	.00	1.18	—
20	20	10	.9782	.072	30	.9851	.3169	.37	.00	1.44	—
20	20	10	.9782	.089	29	.9893	.5108	.65	.03	1.33	3.73
20	20	10	.9782	.072	30	.9868	.3965	.60	.00	1.66	—
20	20	10	.9782	.080	37	.9874	.4236	.62	.00	1.13	—
20	20	10	.9782	.078	34	.9919	.6299	.76	.03	0.86	1.72
20	20	10	.9782	.086	35	.9880	.4486	.77	.00	0.69	—

but the ladder learning runs must be reported separately since each observation depends on the preceding ones. Table 2 summarizes repeated ladder learning runs on problem S. The data are reported in the same format as Table 1.

Table 3 reports the ladder learning distributions of function values by class

TABLE 7

Distributions of Normalized Function Values for QL Class, Problems 13-24, with Learning

Range	Prob. No. 13	14	15	16	17	18	19	20	21	22	23	24
n	50			30			20			10		
m	15	10	5	15	10	5	15	10	5	15	10	5
.995–1.00	—	—	—	—	—	—	—	—	.41	—	—	.33
.990–.995	.15	—	.29	—	.05	.05	—	—	.46	—	.03	.53
.985–.990	.73	.15	.56	.31	.45	.42	.17	.07	.11	.39	.22	.11
.980–.985	.03	.56	.06	.25	.24	.45	.44	.13	.03	.33	.46	—
.975–.980	.03	.21	.06	.17	.11	.03	.22	.42	—	.15	.16	—
.970–.975	.03	.06	—	.14	.08	—	.06	.23	—	.09	.11	—
.965–.970	—	—	—	.06	.05	.03	.08	.13	—	.03	.03	—
.960–.965	—	—	—	—	—	—	.03	—	—	—	—	—
.955–.960	—	—	—	.06	—	—	—	—	—	—	—	.03
.950–.955	—	—	—	—	—	—	—	—	—	—	—	—
.945–.950	—	—	—	—	—	—	—	—	—	—	—	—
.940–.945	—	—	—	—	—	—	—	—	—	—	—	—
.935–.940	—	—	—	—	—	—	—	—	—	—	—	—
.930–.935	—	—	—	—	—	—	—	—	—	—	—	—
.925–.930	—	—	—	—	—	—	—	—	—	—	—	—
.920–.925	—	—	—	—	—	—	—	—	—	—	—	—
.915–.920	—	—	—	—	—	—	—	—	—	—	—	—
.910–.915	—	—	—	—	—	—	—	—	—	—	—	—
.905–.910	—	—	—	—	—	.03	—	—	—	—	—	—
.900–.905	—	—	—	—	—	—	—	—	—	—	—	—
Below .90	.03	.03	.03	.03	.03	—	—	.03	—	—	—	—
N	34	34	34	36	38	38	36	31	37	33	37	36
$^*p$	49	48	48	31	41	50	41	†	50	53	38	50
$^*q$	7	11	6.5	7.5	8	9.5	10	†	4	11	8.5	4
K	.15	.09	.05	.05	.04	.04	.03	†	.03	.05	.05	.06

* Beta parameters are for transformed variable, f'.
† Not available, see Table 8.

interval for the problems summarized in Table 1. On problem 1, for example, a relative frequency of 0.23 was found in the interval of function values from 0.980 to 0.985. For problem 6, 0.05 of the observations were between 0.995 and 1.000. Under each distribution is listed the number of observations taken of the ladder learning type, the parameters of a beta density fit to the empirical distribution, and the value of the Kolmogoroff-Smirnov test statistic for that fit.

Table 4 reports, in the same format as Table 3, distribution data for the repeated runs on problem 8 that were summarized in Table 2.

TABLE 8

Repeated Observations on Distributions of Normalized Function Values for Problem No. 20, QL Class, with Learning

Range	Problem No. 20, *QL*, $n = 20$, $m = 10$																			
.995–1.00	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.990–.995	.14	—	.10	.15	.04	.02	—	—	.03	—	.03	—	—	—	—	—	—	—	.03	—
.985–.990	.67	.42	.63	.63	.48	.26	.21	.14	.47	.17	.37	.23	.20	.22	.03	.17	.07	.11	.29	.32
.980–.985	.13	.40	.15	.12	.38	.57	.53	.66	.36	.17	.37	.29	.23	.47	.27	.48	.40	.46	.44	.34
.975–.980	.04	.07	.06	.03	.03	.05	.12	.08	.03	.42	.13	.33	.30	.19	.23	.10	.23	.14	.06	.17
.970–.975	.01	.06	.03	.01	.01	.03	.05	.05	.04	.15	.05	.07	.10	—	.23	.07	.10	.11	.09	.17
.965–.970	.01	.04	.01	—	.02	—	.05	.03	.03	.07	.05	.07	.10	.03	.13	—	.10	.05	.06	—
.960–.965	—	.01	.01	.01	.01	.03	—	.03	.01	—	—	.03	—	.06	—	.07	—	.05	.03	—
.955–.960	—	—	.01	.01	—	—	—	.01	—	—	—	—	—	—	.07	—	—	.03	—	—
.950–.955	—	—	—	—	—	—	—	—	—	—	—	—	—	—	.03	—	—	—	—	—
.945–.950	—	—	—	.01	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.940–.945	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.935–.940	—	—	—	—	.01	—	.01	—	—	—	—	—	—	—	—	—	.03	—	—	—
.930–.935	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.925–.930	—	—	—	.01	—	—	—	—	—	—	—	—	.03	—	—	—	—	—	—	—
.920–.925	—	—	—	—	—	.02	—	—	—	—	—	—	—	—	—	.03	—	—	—	—
.915–.920	—	.01	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.910–.915	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.905–.910	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.900–.905	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Below .90	—	—	.01	.02	.01	.03	.03	—	.03	.01	—	—	.03	.03	—	.07	.07	.05	—	—
N	142	153	149	150	145	62	66	74	72	69	38	31	30	32	30	29	30	37	34	35
$^*p$	55	53	54	56	48	53	50	54	50	48	53	49	48	†	†	†	†	†	51	48
$^*q$	8.5	11	8.5	9	9	11	11	12	9.5	13	11	12	13	†	†	†	†	†	11	11
K	.08	.07	.08	.08	.05	.04	.06	.10	.06	.07	.04	.04	.09	†	†	†	†	†	.02	.07

* Beta parameters for transformed variable, f'.
† Not available.

TABLE 9

Summary of Output and Problem Data, LL Class, Problems 25–36, Learning and Non-Learning Procedures

Problem Data					Learning Output							Non-Learning Output						
No.	n	m	f_T	t_0	N	f^0	$\%_I$	r_I	r_H	t_I	t_H	N	f^0	$\%_I$	r_I	r_H	t_I	t_H
25	90	15	.9711	.713	29	.9877	.5740	.76	.03	9.98	29.2	77	.9824	.3906	.10	.00	7.13	—
26	90	10	.9711	.713	29	.9929	.7529	.76	.41	9.98	12.1	76	.9827	.4011	.08	.00	9.27	—
27	90	5	.9716	.593	29	.9965	.8782	.69	.55	8.30	8.89	108	.9967	.8843	.08	.05	7.71	13.0
28	60	15	.9701	.235	34	.9880	.6004	.76	.18	3.29	5.64	72	.9812	.3716	.22	.00	1.18	—
29	60	10	.9702	.200	35	.9908	.6930	.80	.34	2.20	5.21	97	.9939	.7961	.17	.03	1.20	6.61
30	60	5	.9704	.180	35	.9954	.8460	.83	.40	1.44	1.44	70	.9916	.7174	.20	.01	1.08	12.8
31	30	15	.9709	.064	38	.9899	.6540	.89	.34	0.57	0.57	76	.9869	.5508	.16	.01	0.44	4.9[illegible]
32	30	10	.9710	.065	36	.9930	.7592	.97	.69	0.52	0.58	641	.9915	.7088	.31	.05	0.26	1.30
33	30	5	.9714	.050	38	.9922	.7269	.99	.87	0.45	0.60	76	.9937	.7797	.18	.04	0.30	1.31
34	10	15	.9707	.013	36	.9903	.6697	.72	.17	0.21	0.37	93	.9828	.4128	.04	.00	0.32	—
35	10	10	.9707	.012	39	.9908	.6862	.87	.20	0.11	0.18	100	.9806	.3392	.02	.00	0.61	—
36	10	5	.9710	.012	37	.9997	.9902	.86	.76	0.10	0.11	92	.9986	.9510	.48	.35	0.04	0.04

TABLE 10

Summary of Output and Problem Data for Repeated Runs on Problem 32, LL Class, Learning

Problem Data					Learning Output						
No.	n	m	f_T	t_0	N	f^0	$\%_I$	r_I	r_H	t_I	t_H
32	30	10	.9710	.042	152	.9959	.8599	.99	.87	0.34	0.59
32	30	10	.9710	.046	146	.9955	.8435	.99	.89	0.46	0.50
32	30	10	.9710	.052	142	.9941	.7973	.96	.82	0.41	0.83
32	30	10	.9710	.051	149	.9942	.8014	1.0	.87	0.41	0.56
32	30	10	.9710	.053	151	.9953	.8381	.98	.89	0.47	0.53
32	30	10	.9710	.056	68	.9955	.8463	.98	.80	0.67	0.78
32	30	10	.9710	.056	75	.9942	.8000	.95	.60	0.51	0.56
32	30	10	.9710	.048	69	.9948	.8204	.96	.88	0.43	0.53
32	30	10	.9710	.052	75	.9946	.8136	.96	.84	0.63	0.83
32	30	10	.9710	.049	75	.9964	.8762	.93	.65	0.44	0.64
32	30	10	.9710	.047	34	.9928	.7510	.88	.53	0.42	0.70
32	30	10	.9710	.060	31	.9925	.7401	.90	.42	0.60	0.90
32	30	10	.9710	.055	32	.9950	.8272	.87	.37	0.44	1.05
32	30	10	.9710	.061	26	.9948	.8218	1.0	.73	0.49	0.55
32	30	10	.9710	.060	36	.9927	.7497	.92	.50	0.48	0.54
32	30	10	.9710	.60	27	.9928	.7537	.96	.56	0.72	0.84
32	30	10	.9710	.061	36	.9942	.8000	.89	.50	0.55	0.61
32	30	10	.9710	.052	33	.9944	.8082	.91	.76	0.46	0.57
32	30	10	.9710	.056	36	.9946	.8136	.92	.67	0.67	0.90
32	30	10	.9710	.053	39	.9927	.7469	.87	.38	0.47	0.69

Tables 5 to 8 for the *QL* class and Tables 9 to 12 for the *LL* class present the same data in the same format for those classes as Tables 1 to 4 for the *QQ* class.

Table 13 reports the distributions of local maxima, the best value located, the sample size, and computer time per iteration for the three problems with given integer solutions. All three problems have 10 variables and two functional con-

TABLE 11

Distributions of Normalized Function Values for LL Class, Problems 25–36 with Learning

Range	Prob. No.											
	25	26	27	28	29	30	31	32	33	34	35	36
	n											
	90			60			30			10		
	m											
	15	10	5	15	10	5	15	10	5	15	10	5
.995–1.00	—	—	.14	—	—	.03	—	—	—	—	—	.16
.990–.995	—	.21	.17	—	.06	.11	—	.53	.08	.03	.05	.54
.985–.990	.10	.24	.28	.21	.31	.26	.34	.17	.18	.17	.21	.05
.980–.985	.24	.21	.07	.26	.29	.20	.32	.17	.26	.33	.39	.03
.975–.980	.28	.10	.03	.18	.06	.23	.10	.06	.13	.11	.13	.03
.970–.975	.14	—	—	.12	.09	—	.13	.06	.21	.08	.10	.05
.965–.970	.03	.03	—	.03	.03	—	.03	—	.05	.08	.03	—
.960–.965	—	—	—	.03	—	—	—	—	—	—	.03	.03
.955–.960	—	—	—	—	—	—	.03	—	.03	.11	—	.08
.950–.955	—	—	—	—	—	—	—	—	.03	.06	.03	.03
.945–.950	—	—	—	—	—	—	.03	—	—	—	.03	—
.940–.945	—	—	—	—	—	—	—	—	—	.03	—	—
.935–.940	—	—	—	—	—	—	—	—	—	—	—	—
.930–.935	.03	—	—	—	—	—	—	—	—	—	—	—
.925–.930	—	—	—	—	—	—	—	—	—	—	—	—
.920–.925	—	.03	.03	—	—	—	—	—	—	—	—	—
.915–.920	—	—	—	—	—	.03	—	—	—	—	.03	—
.910–.915	—	—	—	—	—	—	—	—	—	—	—	—
.905–.910	—	—	—	—	—	—	—	—	—	—	—	—
.900–.905	.03	.03	—	—	—	—	—	—	—	—	—	—
Below .90	.14	.14	.28	.18	.17	.14	.03	.03	.03	—	—	—
N	29	29	29	34	35	35	38	36	38	36	39	37
$^{*}p$	31	24	25	39	35	25	38	†	17	11	25	13
$^{*}q$	9	4.5	3	9.5	7	5	8.5	†	4.5	3.5	6	1.5
K	.02	.03	.06	.03	.03	.02	.05	†	.00	.05	.04	.11

* Beta parameters are for transformed variable, f'.
† Not available, see Table 12.

straints. Problem 37 is of the QQ type, problem 38 of the QL type and problem 39 of the LL type. Note that the true optimum was located in problem 39 and that these distributions are indistinguishable from those reported in the earlier tables.

Some Empirical Results on the Distributions

The observed frequency distributions seem to show a strong regularity from problem to problem. The distributions from problems of different types and from

TABLE 12

Repeated Observations on Distributions of Normalized Function Values for Problem No. 32, LL Class, with Learning

Range	Problem No. 32, *LL*, *n* = 30, *m* = 10																			
.995–1.00	.07	.07	—	—	.01	.02	—	—	—	.07	—	—	—	—	—	—	—	—	—	—
.990–.995	.67	.70	.43	.32	.67	.65	.17	.74	.56	.41	.18	.10	.13	.50	.25	.22	.17	.49	.22	.23
.985–.990	.15	.13	.39	.58	.22	.16	.43	.16	.31	.19	.35	.42	.25	.23	.28	.37	.33	.30	.50	.15
.980–.985	.07	.08	.08	.07	.06	.09	.24	.03	.07	.15	.18	.19	.25	.15	.19	.19	.19	.06	.14	.26
.975–.980	.03	.01	.04	.01	.01	.04	.07	.01	.03	.09	.12	.16	.19	.08	.14	.11	.11	.03	.06	.18
.970–.975	.01	.01	.03	.01	.01	.03	.04	.01	—	.03	.06	.03	.06	.04	.06	.07	.08	.03	—	.05
.965–.970	.01	—	.02	—	—	.01	.01	—	—	.03	.03	.03	.06	—	.03	.04	.03	—	—	.05
.960–.965	—	—	.01	—	.01	—	—	—	.01	.01	—	.03	—	—	—	—	—	—	.03	.03
.955–.960	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.950–.955	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.945–.950	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.940–.945	—	—	—	—	.01	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.935–.940	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.930–.935	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.925–.930	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.920–.925	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.915–.920	—	—	—	—	—	—	—	—	.01	—	—	—	—	—	—	—	—	—	.03	—
.910–.915	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.905–.910	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
.900–.905	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Below .90	—	.01	.01	—	.01	—	.04	.04	.01	.03	.09	.03	.06	—	.06	—	.08	.09	.03	.05
N	152	146	142	149	151	68	75	69	75	75	34	31	32	26	36	27	36	33	36	39
*p	53	57	49	54	57	51	39	60	51	31	30	40	27	39	23	†	†	†	†	†
*q	5.5	5.5	6.5	7	6	5.5	6.5	6	6	4	5.5	7.5	5.5	5	4	†	†	†	†	†
K	.05	.08	.05	.07	.08	.08	.03	.09	.05	.04	.02	.04	.02	.06	.04	†	†	†	†	†

* Beta parameters are for transformed variable, f'.
† Not available.

TABLE 13

Learning Distributions of Normalized Function Values and Problem Data for Problems 37, 38, and 39

Range	$n = 10, m = 2$		
	No.		
	37	38	39
	QQ	*QL*	*LL*
.995–1.00	—	.12	.45
.990–.995	.37	.70	.44
.985–.990	.54	.15	.07
.980–.985	.07	.02	.01
.975–.980	.02	.01	.01
.970–.975	—	—	.01
.965–.970	—	—	—
.960–.965	—	—	.01
.955–.960	—	—	—
.950–.955	—	—	—
.945–.950	—	—	—
.940–.945	—	—	—
.935–.940	—	—	—
.930–.935	—	—	—
.925–.930	—	—	—
.920–.925	—	—	—
.915–.920	—	—	—
.910–.915	—	—	—
.905–.910	—	—	—
.900–.905	—	—	—
Below .90	—	—	—
f^0	.9947	.9970	1.000
N	380	374	390
t_0	.0100	.0074	.0041

different problems of the same type resemble one another. Moreover differences between distributions for different problems seem no larger than the differences between distributions obtained from different runs on the same problem. Tables 3 and 4, 7 and 8, and 11 and 12 show these results.

Beta distributions give good fits to all of the observed distributions. To keep the beta parameter values within limits of available tables the normalized function values f were transformed to $f' = 10.f - 9$ and the fits made to values of f'. We used the Kolmogoroff-Smirnov test applied to the right half of the distribution of f' to examine the goodness of fit. The values of the Kolomogoroff-Smirnov test statistic K are given in Tables 3, 4, 7, 8, 11 and 12. These values suggest the beta distribution gives an excellent fit in every case.

These results suggest that applying our solution procedure to a problem amounts to determining a beta distribution of normalized local optima. This leads

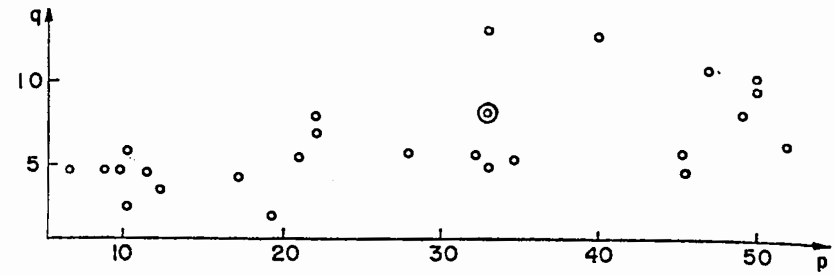

FIG. 2. Fitted beta parameter points, QQ problems

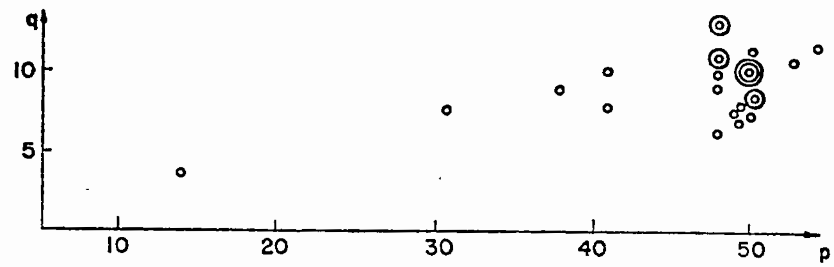

FIG. 3. Fitted beta parameter points, QL problems

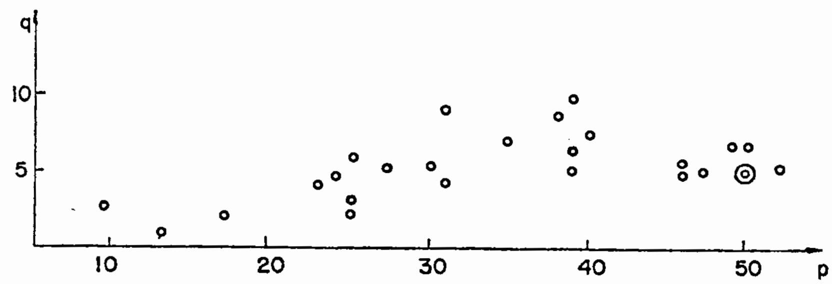

FIG. 4. Fitted beta parameter points, LL problems

us naturally to the idea of looking at the parameters of the beta distributions obtained in our test problems for regularities which might characterize types of problems.

Figures 2, 3 and 4 are scatter plots of the parameter pair (p, q) for the QQ, QL, and LL classes, respectively. These plots show a great deal of regularity in the location of the parameter points for each class, indicated by the high degree of linear correlation between the parameters apparent to the eye. There is, however, little regularity as a function of n, m or N for any class. It looks as though there is no point in distinguishing between QQ, QL and LL problems as far as the distribution of local maxima obtained by our solution procedure is concerned.

Our test results suggest
1. Our solution procedu
in all classes of problems t
2. The approximation o
lem. The distribution of l
classes of problems and for
sampled will give a value
same in all cases.
3. The cost of computir
For any of the three class
increase at an increasing r
rate with m, the number
per iteration is least in the
and largest in the case of
dependent on the numbe
function and constraints.
4. The use of "learning
tained for a given cost of
found by ladder learning
learning observations. In
size was no more than al
$7 f^0$ is much greater for
from f_T to f^* was signi
Ladder learning iteratio
the problems while no
lems. The reported valu
learning than for learni
servation and is therefo
servations required bef
values above f_T but that
Choosing the learning
have little to say about
these values did produc
some problems.
5. Our test computa
optimal" integer points
solution procedures pro
small fraction of the opt
scattered in the feasib
number of approxima
values but perhaps ver
option of choosing an
formal problem.

Conclusions

Our test results suggest the following conclusions:

1. Our solution procedures give good approximations to the integer solutions in all classes of problems tested.

2. The approximation obtained does not seem to depend on the type of problem. The distribution of local maxima obtained is about the same in all three classes of problems and for all sizes of problems; the likelihood that the best point sampled will give a value within $100.\alpha\%$ of the optimum, for any α, is about the same in all cases.

3. The cost of computing does depend heavily on the type and size of problem. For any of the three classes of problem computer time per iteration appears to increase at an increasing rate with n, the number of variables, and at a decreasing rate with m, the number of constraints. As one would expect, the computer time per iteration is least in the case of LL problems, larger in the case of QL problems, and largest in the case of QQ problems. Computer time per iteration seems mainly dependent on the number of multiplications required to evaluate the objective function and constraints.

4. The use of "learning" results in substantial improvement in the results obtained for a given cost of computing. For 30 of the 36 problems the best point, f^0, found by ladder learning iterations was greater than the best point found on nonlearning observations. In every problem but number 7 the ladder learning sample size was no more than about half of the nonlearning sample size and in problem 7 f^0 is much greater for ladder learning. The relative frequency of the interval from f_T to f^* was significantly greater for ladder learning on every problem. Ladder learning iterations found values in the upper half of the interval for 29 of the problems while nonlearning iterations were that successful on only 14 problems. The reported values of t_I, the time to reach f_T, are generally lower for nonlearning than for learning. However, the value of t_I for learning is an actual observation and is therefore inflated by the time used by the six nonlearning observations required before learning begins. These six observations could reach values above f_T but that event is not reflected in t_I for learning.

Choosing the learning parameters q and r is very much an open question. We have little to say about that, having used only the values 0.75 and 6, except that these values did produce our results, and results of this quality may be useful in some problems.

5. Our test computations reveal something about the ubiquity of "nearly optimal" integer points. In the fairly large test problems of all three types, our solution procedures produced many lattice points with function values within a small fraction of the optimal value. Such points may be, and generally are, widely scattered in the feasible region. Thus our procedure typically gives rise to a number of approximations to the solution, with about equally good function values but perhaps very different coordinates. The problem solver then has the option of choosing among these on the basis of considerations external to the formal problem.

6. A "real" application of our solution procedure requires the user to choose a sampling plan; he must somehow decide when to stop sampling local maxima and announce an approximation to the solution. Sampling should stop when the expected return from one more observation is less than the cost of taking one more observation.[2]

References

1. Dantzig, George, *Linear Programming and Extensions*, Princeton University Press, Princeton, New Jersey, 1963, Chapter 26, pp. 535–550.
2. Reiter, Stanley and Sherman, Gordon, "Discrete Optimizing," *Journal of the Society for Industrial and Applied Mathematics*, Volume 13, No. 3, September 1965.
3. Rosen, J. and Suzuki, S., "Construction of Nonlinear Programming Test Problems," *Comm. of ACM*, Volume 8, No. 2, February 1965.

[2] The choice of a sequential sampling plan is discussed in [1], where Bayes' optimal and other sequential sampling plans are given for some cases.