J. Austral. Math. Soc. **21** (Series B) (1980), 345–356

AN ALGORITHM FOR SOLVING THE RESTRICTED LEAST SQUARES PROBLEM

DAVID CLARK

(Received 10 November 1978)

(Revised 31 January 1979)

Abstract

This paper presents an algorithm to solve the least squares problem when the parameters are restricted to be non-negative. The algorithm is based on the branch and bound method which has been suggested for this problem, and shares with it the property that an unrestricted least squares subproblem is solved at each step. However, improvements have been made to the branching rules by making use of the convexity of the problem, and the Kuhn–Tucker conditions are used to test for optimality. The resulting algorithm becomes essentially iterative in nature, and linearity of the number of subproblems solved can be shown under assumptions which have always been observed in practice.

1. Introduction

1.1. THE PROBLEM. The problem considered here is the restricted least squares problem:

$$\begin{aligned} &\text{Minimize} \quad L(\lambda) = (\mathbf{y} - X\lambda)^{\mathrm{T}}(\mathbf{y} - X\lambda), \\ &\text{subject to} \quad \lambda > \mathbf{0}, \end{aligned} \tag{P}$$

where X is an $m \times n$ matrix, $\mathbf{y} \in R^m$, $\lambda \in R^n$. The problem P can be solved by converting it into a linear programming problem [3]. Another approach, suggested by Waterman [4], is to solve a sequence of up to 2^n unrestricted problems. Beale [2] considers it as an example of his optimum subset selection algorithm using partial enumeration. Armstrong and Frome [1] place it in a branch and bound framework and give an improved pruning rule.

The problem, however, is a very special one in that provided X is of full rank, the minimization is of a strictly convex function over a convex set, so that the solution is unique. Advantage is taken of the special properties of the problem to

develop an algorithm which is essentially iterative, and which displays an apparent linear increase in the number of subproblems solved as the dimension of X increases.

The remainder of the paper follows the development of the algorithm which starts from the branch and bound approach. The rest of Section 1 defines notation and introduces the branch and bound method. In Section 2, the Armstrong and Frome algorithm is presented and an improved pruning rule given. Then the Kuhn–Tucker conditions for optimality are established. A rule is given to find a better feasible solution should a feasible solution be found to be sub-optimal. The complexity of the algorithm is considered in Section 3 and, under certain assumptions (always satisfied experimentally), linearity of subproblems solved against problem dimension is proved. The experimental results are presented in Section 4. In Section 5, the extension of the algorithm to similar problems is discussed, including the modifications necessary if the Kuhn–Tucker conditions are not readily available. Section 6 contains the concluding remarks.

1.2. NOTATION. The following notation will be used in the paper:

J^i represents an index set $J^i \subseteq N = \{1, 2, \ldots, n\}$.

P^i represents the problem:

Minimize $L(\boldsymbol{\lambda})$,

subject to $\lambda_j = 0, j \in J^i$.

$\boldsymbol{\lambda}^i$ represents the optimal solution to P^i.

(Note that the index set J^i defines P^i and hence $\boldsymbol{\lambda}^i$.)

$\boldsymbol{\mu}^i$ represents a feasible solution to both P and P^i, that is, $\boldsymbol{\mu}^i \geqslant \mathbf{0}$ and $\mu^i_j = 0$, $j \in J^i$.

(Note that $\boldsymbol{\mu}^i$ will not necessarily be optimal for P or P^i.)

When the term "feasible" is used, it will refer to feasibility for P, that is, $\boldsymbol{\lambda}$ is feasible if $\boldsymbol{\lambda} \geqslant \mathbf{0}$.

1.3. BRANCH AND BOUND. The branch and bound method builds up a search tree (each node being a problem, P^i) by increasing the number of variables set to zero as a branch is descended. Thus, if P^j is a descendant of P^i, $J^j \supset J^i$. The root of the tree is the problem P^1 where $J^1 = \emptyset$. An example of a search tree is given in Fig. 1.

The main considerations of a branch and bound algorithm are:

(i) Choosing which node to branch on next,

(ii) choosing which descendant of this node to consider (solve) next, and

(iii) making use of any special properties of the problem to detect early fathoming of a branch, that is, recognizing when no descendants of a node will yield a better solution.

All of the above considerations are dealt with in the new algorithm.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

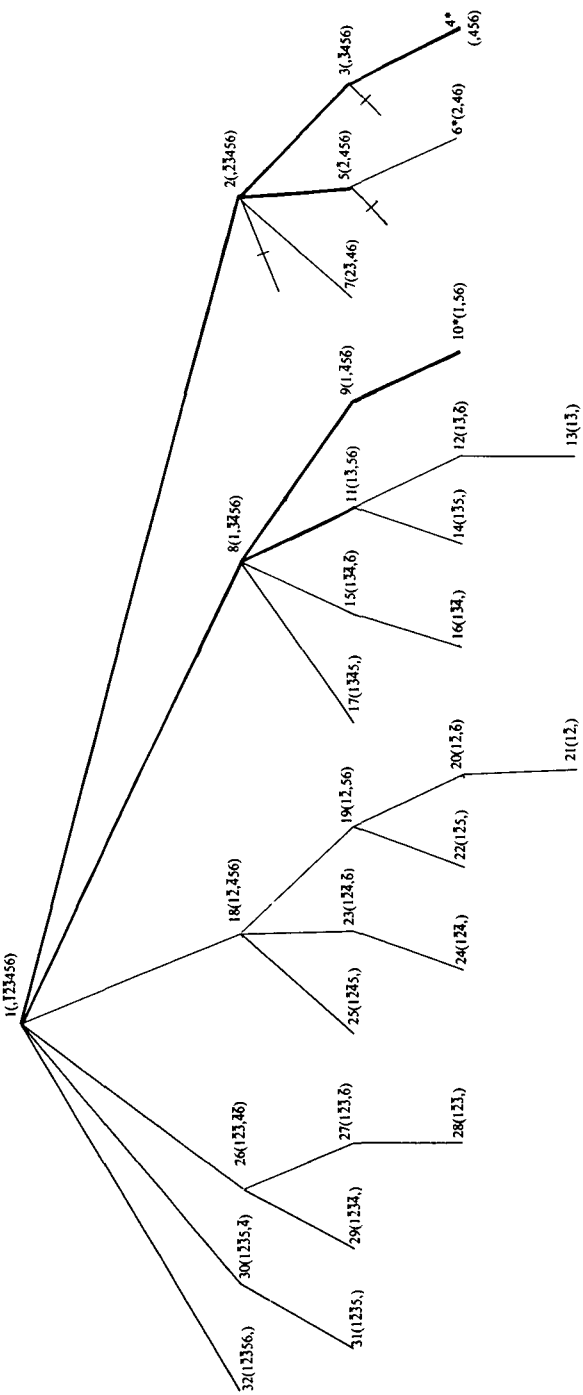

Fig. 1. A solution tree for the sample problem. At each node the numbers indicate which variables were used in the subproblem, those after the comma being optional. A − indicates that the variable is negative and an * indicates the solution is feasible (that is, $\lambda \geqslant 0$). The whole tree is generated by the Armstrong–Frome algorithm. The thickened lines represent the tree generated using the improved pruning rule of Theorem 1.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

2. The new algorithm

2.1. The Armstrong and Frome algorithm. Armstrong and Frome's node choice is to branch on the node most recently solved until a feasible λ^i is found, and thereafter to branch on the node, P^j, with the smallest $L(\lambda^j)$. Their choice of the next variable to be set to zero is the most negative free variable if one exists, otherwise the free variable with the largest numerical value. Their pruning rule states that if a node differs from its parent node in that a variable which was negative in the parent node's optimal solution has been set to zero (for example, nodes 2, 8, 18, and 26, 30, 32 in Figure 1), and either the optimal solution of the node is feasible (for example, nodes 4, 6, 10) or has an $L(\lambda)$ greater than or equal to the best existing feasible solution (nodes 6, 7), then no further branches from the present node need be considered. In the example of Fig. 1 (data in Table 1, results in Table 2), 32 of the possible 64 nodes were solved.

TABLE 1

Data for the sample problem

X_1	X_2	X_3	X_4	X_5	X_6	Y
1.00	4.70	7.89	7.93	3.47	8.35	6.94
1.00	3.10	3.46	5.35	2.97	7.11	5.77
1.00	8.34	6.68	1.75	8.68	8.90	8.04
1.00	4.62	2.69	9.20	5.39	1.60	5.12
1.00	1.03	6.22	6.25	4.75	3.61	7.82
1.00	3.26	5.64	9.10	6.53	4.70	13.26
1.00	2.27	5.34	5.15	7.27	3.16	13.47
1.00	7.27	3.64	6.65	7.77	3.78	12.49
1.00	5.93	6.65	8.65	9.77	0.92	11.06
1.00	0.47	0.45	1.63	1.90	8.66	14.40

Column 1 contains 1.00 because the model is: $Y = \lambda_1 + \sum_{i=2}^{6} \lambda_i X_i$, not $Y = \sum_{i=1}^{6} \lambda_i X_i$.

2.2. Improved pruning rule. The first improvement to the above algorithm is the new fathoming criterion "at node P^i, it is only necessary to branch on variables λ_j for which $\lambda_j^i < 0$".

Lemma 1. *Given $\mu^i \geqslant \mathbf{0}$. Let $J^i = \{j \mid \mu_j^i = 0\}$ define P^i and hence λ^i. Then there exists some descendant, P^r, of P^i (that is, $J^r \supseteq J^i$) for which $\lambda^r \geqslant \mathbf{0}$ and $L(\lambda^r) \leqslant L(\mu^i)$.*

Proof. If $\lambda^i \geqslant \mathbf{0}$, $\lambda^r = \lambda^i$; otherwise let k be such that

$$\frac{\mu_k^i}{|\lambda_k^i|} = \min\left\{\frac{\mu_j^i}{|\lambda_j^i|} \mid \lambda_j^i < 0\right\}.$$

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

TABLE 2

Solutions obtained on sample program using Armstrong/Frome algorithm

Node	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	$L(\lambda)$
1	−7.27	−1.89	−1.34	0.92	2.91	1.70	32.09
2	0	−1.52	−1.05	0.44	2.23	1.08	36.42
3	0	0	−0.95	0.49	1.21	0.84	102.00
4	0	0	0	0.25	0.84	0.62	127.18
5	0	−1.45	0	0.18	1.78	0.84	66.72
6	0	0.04	0	0.81	0	0.80	184.66
7	0	0.05	−0.02	0.82	0	0.81	184.65
8	10.22	0	−0.58	−0.20	0.59	0.04	86.07
9	13.31	0	0	−0.53	0.22	−0.30	93.51
10	7.52	0	0	0	0.33	0.08	103.49
11	8.10	0	−0.70	0	0.69	0.21	87.04
12	11.96	0	−0.35	0	0	−0.09	102.86
13	11.49	0	−0.34	0	0	0	103.45
14	9.73	0	−0.63	0	0.54	0	89.47
15	15.84	0	−0.17	−0.51	0	−0.40	94.45
16	12.34	0	−0.26	−0.20	0	0	100.99
17	10.67	0	−0.55	−0.23	0.56	0	86.11
18	4.43	−1.25	0	−0.07	1.44	0.50	64.25
19	3.57	−1.29	0	0	1.49	0.56	64.39
20	11.42	−0.29	0	0	0	−0.08	103.42
21	11.00	−0.28	0	0	0	0	103.91
22	8.29	−0.90	0	0	0.90	0	78.00
23	16.34	−0.24	0	−0.55	0	−0.42	92.22
24	12.39	−0.26	0	−0.24	0	0	99.92
25	10.11	−0.93	0	−0.37	1.00	0	68.90
26	16.37	−0.22	−0.08	−0.52	0	−0.41	92.01
27	12.38	−0.20	−0.26	0	0	−0.09	100.75
28	11.90	−0.20	−0.25	0	0	0	101.36
29	12.78	−0.21	−0.17	−0.20	0	0	98.82
30	10.83	−0.88	−0.47	−0.28	1.14	0	61.56
31	9.69	−0.85	−0.57	0	1.10	0	66.50
32	4.05	−1.34	−0.76	0	1.93	0.73	45.04

Let $J^{i+1} = J^i \cup \{k\}$ define P^{i+1} and λ^{i+1}. Let

$$\mu^{i+1} = \frac{\mu_k^i}{\mu_k^i + |\lambda_k^i|}\lambda^i + \frac{|\lambda_k^i|}{\mu_k^i + |\lambda_k^i|}\mu^i \geqslant 0.$$

Then, from the convexity of L and the optimality of λ^i and λ^{i+1},

$$L(\lambda^{i+1}) \leqslant L(\mu^{i+1}) \leqslant L(\mu^i).$$

If $\lambda^{i+1} \geqslant 0$, $\lambda^r = \lambda^{i+1}$. Otherwise, the process is repeated and, at each step,

$$L(\lambda^{i+J}) \leqslant L(\mu^{i+J}) \leqslant L(\mu^{i+J-1}) \leqslant \ldots \leqslant L(\mu^i),$$

until eventually $\lambda \geqslant 0$.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

The improved pruning rule now follows.

THEOREM 1. *At any node P^i, it is only necessary to branch on variables λ_j for which $\lambda_j^i < 0$ in order to find λ^r, a best feasible solution descendant from λ^i.*

PROOF. Let $J_-^i = \{j \mid \lambda_j^i < 0\}$.

Let λ^r be the best feasible descendant of λ^i and assume it could not be reached through a branch in which some λ_j, $j \in J_-^i$, was set to zero, that is,

$$\lambda_j^r > 0 \quad \text{for all } j \in J_-^i.$$

Then there will exist $\mu^i \geqslant \mathbf{0}$, which is a convex linear combination of λ^r and λ^i, which is feasible for P^i. As $L(\lambda^i) < L(\lambda^r)$, it follows that $L(\mu^i) < L(\lambda^r)$.

So, by Lemma 1, there exists λ^s, a descendant of λ^i, for which $\lambda^s \geqslant \mathbf{0}$ and $L(\lambda^s) \leqslant L(\mu^i) < L(\lambda^r)$. Moreover, the method used in the proof of Lemma 1 only ever set $\lambda_j^i < 0$ to zero in P^{i+1}. Hence a best feasible solution descendant from λ^i can be found by branching on only negative-valued variables at any node.

In the example given in Fig. 1, the tree generated using the pruning rule is shown by the thickened lines. The number of subproblems solved has been reduced from 32 to 11. However, tests done using this rule showed that the number of subproblems solved still rose exponentially with problem dimension. The main cause of the exponential rise in the work done appears to be the need to check all branches until the fathoming criteria are satisfied, to ensure the optimum has been found, although in each case tried the actual optimum was found early in the calculation.

2.3. OPTIMALITY CONDITIONS. Once a feasible solution has been found, the Kuhn–Tucker conditions can be used to test its optimality.

THEOREM 2. *If $\lambda^r \geqslant \mathbf{0}$ solves P^r, and $X^{\mathrm{T}} X \lambda^r - X^{\mathrm{T}} \mathbf{y} \geqslant \mathbf{0}$, then λ^r solves P.*

(Note that X is the full data matrix, not that part of it used in solving P^r. Of necessity, $X_r^{\mathrm{T}} X_r \lambda^r - X_r^{\mathrm{T}} \mathbf{y}_r = \mathbf{0}$, where X_r is obtained from X by deleting those columns corresponding to indices in J^r.)

PROOF. For the general problem:

$$\begin{aligned} &\text{Minimize} \quad f(\mathbf{x}), \\ &\text{subject to} \quad \mathbf{g}(\mathbf{x}) \geqslant \mathbf{0}, \end{aligned}$$

where $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_m(\mathbf{x}))^{\mathrm{T}}$, the Kuhn–Tucker necessary conditions for optimality are:

(i) $\mathbf{g}(\mathbf{x}) \geqslant \mathbf{0}$,

(ii) there exists $\mathbf{u} \geqslant \mathbf{0}$ such that $\nabla f(\mathbf{x}) = \nabla \mathbf{g}(\mathbf{x})\mathbf{u}$, and

(iii) $\mathbf{u}^{\mathrm{T}} \mathbf{g}(\mathbf{x}) = 0$.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

Moreover, where the objective function is convex and the constraints are linear, as in P, the conditions are also sufficient. Here, we have

$$\mathbf{g}(\boldsymbol{\lambda}) = \boldsymbol{\lambda},$$
$$\nabla \mathbf{g}(\boldsymbol{\lambda}) = I \text{ and,}$$
$$\nabla L(\boldsymbol{\lambda}) = 2(X^{\mathrm{T}} X\boldsymbol{\lambda} - X^{\mathrm{T}}\mathbf{y}).$$

Let $\mathbf{u}^r = \nabla L(\boldsymbol{\lambda}^r) \geqslant \mathbf{0}$; then

$$\nabla L(\boldsymbol{\lambda}^r) = \mathbf{u}^r = \nabla \mathbf{g}(\boldsymbol{\lambda}^r)\mathbf{u}^r.$$

Also, as $\boldsymbol{\lambda}^r$ solves P^r,

$$u_i^r = \nabla L(\boldsymbol{\lambda}^r)_i = 0 \quad \text{for } i \notin J^r,$$

but

$$\lambda_i^r = 0 \qquad \text{for } i \in J^r;$$

hence

$$\mathbf{u}^{r\,\mathrm{T}} \boldsymbol{\lambda}^r = 0.$$

So $(\boldsymbol{\lambda}^r, \mathbf{u}^r)$ is the Kuhn–Tucker point for P. Hence $\boldsymbol{\lambda}^r$ solves P.

2.4. Selection of the next node. If the above test for optimality fails, it can still be used to determine the next node to branch on, and which branching variable should be chosen at that node.

Theorem 3. *If $\boldsymbol{\lambda}^r \geqslant \mathbf{0}$ solves P^r, $\mathbf{u}^r = \nabla L(\boldsymbol{\lambda}^r)$, $u_k^r < 0$ for some $k \in J^r$, and $J^{r+1} = \{i \mid \lambda_i^r = 0,\ i \neq k\}$, then there is some descendant, $\boldsymbol{\lambda}^s$, of $\boldsymbol{\lambda}^{r+1}$ which is feasible and for which $L(\boldsymbol{\lambda}^s) < L(\boldsymbol{\lambda}^r)$.*

Proof. Using the well-known convex function property

$$f(\mathbf{x}_2) \geqslant f(\mathbf{x}_1) + \nabla f(\mathbf{x}_1)^{\mathrm{T}}(\mathbf{x}_2 - \mathbf{x}_1),$$

we have

$$L(\boldsymbol{\lambda}^{r+1}) \geqslant L(\boldsymbol{\lambda}^r) + \mathbf{u}^{r\,\mathrm{T}}(\boldsymbol{\lambda}^{r+1} - \boldsymbol{\lambda}^r).$$

But $L(\boldsymbol{\lambda}^{r+1}) < L(\boldsymbol{\lambda}^r)$, as either P^{r+1} is less restricted than P^r ($J^{r+1} \subset J^r$), or else, if $\lambda_i^r = 0$ for some $i \notin J^r$, then $\boldsymbol{\lambda}^r$ also solves $P^{r'}$, where $J^{r'} = J^r \cup \{i\}$, and again $J^{r+1} \subset J^{r'}$. Thus

$$0 > \mathbf{u}^{r\,\mathrm{T}}(\boldsymbol{\lambda}^{r+1} - \boldsymbol{\lambda}^r) = \mathbf{u}^{r\,\mathrm{T}} \boldsymbol{\lambda}^{r+1} = u_k^r \lambda_k^{r+1}.$$

Hence

$$\lambda_k^{r+1} > 0.$$

Hence there exists $\boldsymbol{\mu}^{r+1} \geqslant \mathbf{0}$, which is a convex linear combination of $\boldsymbol{\lambda}^{r+1}$ and $\boldsymbol{\lambda}^r$ and so $L(\boldsymbol{\mu}^{r+1}) < L(\boldsymbol{\lambda}^r)$, and which is feasible for P^{r+1}. The proof now follows from Lemma 1.

One point worth noting is the definition of J^{r+1} above. It could not be defined as $\{j \mid j \in J^r, j \neq k\}$ as it may be that $\lambda_i^r = 0$, for some $i \notin J^r$, and, if $\lambda_i^{r+1} < 0$, then no convex linear combination, $\boldsymbol{\mu}^{r+1}$, of $\boldsymbol{\lambda}^r$ and $\boldsymbol{\lambda}^{r+1}$ can have $\mu_i^{r+1} \geqslant 0$.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

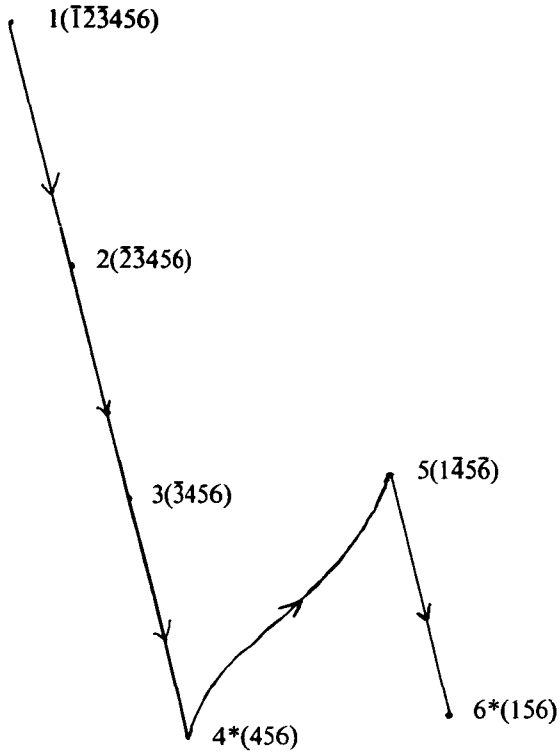

Fig. 2. The solution tree for the sample problem using the new algorithm. The numbers in parentheses represent the variables used in the solution, those with a $-$, being negative. An * indicates that the solution is feasible (that is, $\lambda \geqslant 0$).

TABLE 3

Solutions obtained on sample problem using new algorithm

Node	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	$L(\lambda)$
1	−7.27	−1.89	−1.34	0.92	2.91	1.70	32.09
2	0	−1.52	−1.05	0.44	2.23	1.08	36.42
3	0	0	−0.95	0.49	1.21	0.84	102.00
4	0	0	0	0.25	0.84	0.62	127.18
5	13.31	0	0	−0.53	0.22	−0.30	93.51
6	7.52	0	0	0	0.33	0.08	103.49

Figure 2 shows that portion of the tree generated using Theorems 2 and 3 for the test problem of Fig. 1. The nodes generated are given in Table 3. The first four nodes correspond to nodes 1 to 4 of the earlier tree, and nodes 5 and 6 correspond to nodes 9 and 10 respectively.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

It should be mentioned here that the example was chosen for its illustrative properties rather than its typicality. In over 90% of the test problems solved, the first feasible solution found was the optimum, and in the majority of the rest the algorithm jumped directly to the optimum node.

2.5. THE ALGORITHM SUMMARIZED.

1. Solve the unrestricted problem P^1 with $J^1 = \emptyset$. If $\boldsymbol{\lambda} \geqslant \mathbf{0}$, stop; otherwise, set $i \leftarrow 1$ and go to 2.
2. Set $i \leftarrow i+1$. Use some heuristic to select k from the set $\{j \mid \lambda_j^{i-1} < 0\}$ and let $J^i = J^{i-1} \cup \{k\}$. Solve P^i. If $\boldsymbol{\lambda}^i \geqslant \mathbf{0}$, go to 3; otherwise go to 2.
3. Use theorem 2 to test $\boldsymbol{\lambda}^i$ for optimality. If the test succeeds, stop; otherwise, go to 4.
4. Use some heuristic to choose k from the set $\{j \mid u_j^i < 0\}$, where $\mathbf{u}^i = \nabla L(\boldsymbol{\lambda}^i)$; set $i \leftarrow i+1$, and let $J^i = \{j \mid \lambda_j^{i-1} = 0, j \neq k\}$. Solve P^i. If $\boldsymbol{\lambda}^i \geqslant \mathbf{0}$, go to 3; otherwise, go to 5.
5. Set $i \leftarrow i+1$. Choose k according to the method used in the proof of Lemma 1. Let $J^i = J^{i-1} \cup \{k\}$. Solve P^i. If $\boldsymbol{\lambda} \geqslant \mathbf{0}$, go to 3; otherwise, go to 5.

The heuristics used in steps 2 and 4 were to choose the most negative variable in each case (but see Section 4 for a fuller discussion).

3. Complexity

It appears difficult to determine any absolute complexity bounds for the algorithm, but if the assumption is made that, at any feasible $\boldsymbol{\lambda}^r$ which fails the optimality test of Theorem 2, it does so for only one variable, then linear bounds can be derived.

THEOREM 4. *If $\boldsymbol{\lambda}^r \geqslant \mathbf{0}$ solves P^r, $\mathbf{u}^r = \nabla L(\boldsymbol{\lambda}^r)$, and $u_i^r \geqslant 0$ for $i \in J^r - \{k\}$, then N, the number of subproblems solved, is at most $2n$.*

PROOF. Let

$$J^{r+i} = J^r - \{i\} \quad \text{for all } i \in J^r - \{k\}.$$

Then, by an argument similar to that used in proving Theorem 3,

$$\lambda_i^{r+i} < 0 \quad \text{for all } i \in J^r - \{k\}.$$

Now for any $\boldsymbol{\lambda}^s \geqslant \mathbf{0}$ such that $L(\boldsymbol{\lambda}^s) < L(\boldsymbol{\lambda}^r)$, assume that $\lambda_k^s = 0$.

Let

$$J' = \{j \mid \lambda_j^s > 0, j \in J^r\}.$$

Then there will exist a convex linear combination, $\boldsymbol{\mu}^r \geqslant \mathbf{0}$, of $\boldsymbol{\lambda}^r$, $\boldsymbol{\lambda}^s$ and $\boldsymbol{\lambda}^{r+i}$, $i \in J'$, which is feasible for P^r and for which $L(\boldsymbol{\mu}^r) < L(\boldsymbol{\lambda}^r)$, which contradicts the

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

optimality of λ^r. Hence for each subsequent feasible solution, λ^s, found after λ^r, $\lambda_k^s > 0$.

Now this applies at each step, so that after each feasible solution is found by the algorithm, one more variable must remain strictly positive. Thus, if α_i is the number held to zero in the ith feasible solution found, then $\alpha_i \leqslant n+1-i$, and also there will be at most n feasible solutions found.

Let f_i be the number of subproblems solved between the $(i-1)$st (exclusive) and ith (inclusive) feasible subproblems, and F the total number of feasible solutions found. Evidently, for $i > 1$, $f_i = \alpha_i - \alpha_{i-1} + 2$, and if α_0 is defined as 1, the formula is also correct for f_1.

Thus

$$\begin{aligned} N &= \sum_{i=1}^{F} f_i \\ &= \sum_{i=1}^{F} \alpha_i - \alpha_{i-1} + 2 \\ &= 2F + \alpha_F - \alpha_0 \\ &\leqslant 2F + n + 1 - F - 1 \\ &\leqslant 2n. \end{aligned}$$

Although there are no *a priori* grounds for supposing that the above assumptions are always true, no case has yet been found in which the assumption did not hold. Indeed, the example of Fig. 1, with $N = n$, was the only instance of $N/n \geqslant 1$ (see Table 4).

4. Results

The algorithm was tested against data generated using a random number generator. For each problem size, ten sets of data were solved. Due to lack of consistency of execution times, N, the number of subproblems solved, was taken as the measure of algorithm efficiency. (As an indication, however, solving problems with 40 variables and 50 rows of data took 5 to 10 seconds on a Univac 1110/42.) Armstrong and Frome claimed competitiveness for their algorithm, and as it was the progenitor of the new algorithm, it was used for comparison purposes. The results, given in Table 4, display the linearity predicted in Section 3.

Of the several heuristics tested for choosing the variable to be set to zero at step 2 of the algorithm, choosing the most negative and choosing the negative variable which had differed least from λ^1 proved best. The former was chosen for its simplicity. No choice ever had to be made at step 4, but choosing the most negative u_i is suggested.

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

TABLE 4

Experimental results. Number of subproblems solved

Dimension of X		Armstrong/Frome algorithm		Armstrong/Frome improved		New algorithm		New algorithm modified	
n	m	Mean	Worst	Mean	Worst	Mean	Worst	Mean	Worst
6	10	10.0	32	4.4	11	3.3	6	4.9	13
10	15	168.2	566	34.6	96	5.4	8	8.8	14
15	20	4097.8	11886	668.6	1947	10.0	13	18.0	24
20	30	—	—	—	—	11.9	14	21.8	26
30	40	—	—	—	—	16.6	19	31.2	36
40	50	—	—	—	—	24.4	28	46.8	54

5. Extension to other problems

The features of the restricted least squares problem which make it suitable for the algorithm as given are:

(i) the strict convexity of the objective function;

(ii) the special nature of the constraints; and

(iii) the ease with which each subproblem P^i can be solved.

Any problem which has the above properties is suitable for solving by the algorithm. One question which arises in other applications is the optimality test if the Kuhn–Tucker conditions are not readily available, as this test is central to the algorithm. If this is the case, Theorem 2 can be replaced by one which requires solving no more than $n-1$ additional subproblems to test the optimality of a feasible solution to a subproblem.

THEOREM 5. *Let* $\lambda^r \geqslant 0$ *solve* P^r. *Define* $J^{r+i} = J^r - \{i\}$ *for all* $i \in J^r$. If $\lambda_i^{r+i} < 0$ *for all* $i \in J^r$, *then* λ^r *solves* P.

PROOF. Assume the above conditions hold and further assume there exists λ' such that $L(\lambda') < L(\lambda^r)$. Now, for $i \in J^r$, $\lambda_i' \geqslant 0$, and for some $i \in J^r$, $\lambda_i' > 0$. But, for $i \in J^r$, $\lambda_i^{r+i} < 0$. Hence there is a convex linear combination, λ^s, of λ' and λ^{r+i} for all $i \in J^r$ for which $\lambda_i^s = 0$ for all $i \in J^r$, so that λ^s is feasible for P^r.

Now $L(\lambda^s) \leqslant$ convex linear combination $(L(\lambda'), L(\lambda^{r+i})$ for all $i \in J^r)$. Since $L(\lambda') > L(\lambda^r)$, $L(\lambda^{r+1}) \leqslant L(\lambda^r)$ and the contribution of λ' to λ^s is non-zero, it follows that $L(\lambda^s) < L(\lambda^r)$, which contradicts the assumption that λ^r solves P^r. Hence λ^r solves P.

The only modifications to the algorithm necessary are to replace Theorem 2 with Theorem 5 in step 3, and to omit step 4.

Under the assumptions of Theorem 4 (the complexity theorem), it is easy to

https://doi.org/10.1017/S0334270000002435 Published online by Cambridge University Press

show that the number of subproblems solved is not more than $\frac{1}{2}n(n+1)$. However, testing the modified algorithm on the same test data as used before indicated that in practice this algorithm also behaves linearly (see Table 4).

6. Conclusion

The algorithm presented here appears to be a considerable improvement on existing algorithms of branch and bound type for this problem, without sacrificing any of the advantages of these algorithms, for example, ease of use in an interactive mode, wide availability of least squares regression routines, and simple modification to account for variable bounds.

The reason would appear to be that in this problem, as so often in branch and bound, the optimum solution is found quickly and then much time is spent in the subsequent searching necessary to verify it. Thus the biggest advantage of this approach is in the use of optimality conditions to improve bounding. Incorporating this into the general framework of branch and bound has resulted in a very efficient algorithm.

Acknowledgements

The author is indebted to Dr. M. R. Osborne and Dr. R. B. Stanton for helpful discussions on the algorithm and for advice on the presentation of the paper. The author also wishes to thank the referees for suggestions which have improved the presentation of the paper.

References

[1] R. D. Armstrong and E. L. Frome, "A branch-and-bound solution of a restricted least squares problem", *Technometrics* 18 (1976), 447–450.

[2] E. M. L. Beale, "Selecting an optimum subset", Chapter 22 of *Integer and non-linear programming* (ed. J. Abadie) (Amsterdam: North Holland Publishing Co., 1970).

[3] G. G. Judge and T. Takayama, "Inequality restrictions in regression analysis", *J. Amer. Statist. Assoc.* 61 (1966), 166–181.

[4] M. S. Waterman, "A restricted least squares problem", *Technometrics* 16 (1974), 135–136.

Computing Research Group
Australian National University
Canberra
A.C.T. 2600