A PARAMETRIC BOUNDING METHOD FOR FINDING A MINIMUM ℓ_∞-NORM SOLUTION TO A SYSTEM OF EQUATIONS

by

J. A. Tomlin

TECHNICAL REPORT SOL 75-12

May 1975

SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
Stanford University
Stanford, California

Research and reproduction of this report were partially supported by the Office of Naval Research under Contract N00014-75-C-0865 and N00014-75-C-0267; U.S. Atomic Energy Commission Contract AT(04-3)-326 PA #18; and U.S. Army Research Office Contract DAHC04 74 C 0034.

Reproduction in whole or in part is permitted for any purposes of the United States Government. This document has been approved for public release and sale; its distribution is unlimited.

A PARAMETRIC BOUNDING METHOD FOR FINDING A MINIMUM ℓ_∞-NORM SOLUTION TO A SYSTEM OF EQUATIONS

by

J. A. Tomlin

ABSTRACT

This paper presents a method for finding the minimum ℓ_∞-norm solution to a set of consistent linear equations using a form of parametric linear programming. In this application the upper and lower bounds of all the variables are parameterized, and we work with only the original variables and constraints. Computational results indicate that the method is superior to both a primitive linear programming approach to the problem and to other, more specialized methods, which have been suggested.

1. Introduction

The problem of finding the minimum ℓ_∞-norm solution to a set of consistent linear equations has been recently treated by Cadzow [3], who devised a special purpose algorithm which he claimed was superior to the usual linear programming "formulization."[1]

[1] For some reason Cadzow felt it necessary to invent this appalling word rather than use the perfectly acceptable term "formulation." It is to be hoped that his example will not be followed.

In this claim he is almost certainly correct, but unfortunately both Cadzow and others who have treated this problem (for example in the control theory literature, see e.g. [7]) have used primitive linear programming formulation techniques, apparently based on obsolete text-book methods, which result in unnecessarily inefficient implementations. By using the now standard techniques for dealing implicitly with simple bounds on variables and combining them with the parametric programming approach it is possible to produce a far more efficient direct primal method for solving the problem.

We wish to find a solution to the system of equations

$$Ax = b \tag{1}$$

where A is $m \times n$, of rank m with $m < n$, such that the variable x_j of largest magnitude is as small as possible. This is equivalent to saying we require the minimum ℓ_∞-norm solution as defined by

$$\|x\|_\infty = \max_j |x_j| \qquad (j = 1, \ldots, n) .$$

One approach to this problem is to dualize it [3] and solve the dual problem. Another is to state it as a linear programming (L.P.) problem:

$$\text{Min } \theta \tag{2}$$

$$Ax = b$$

$$-\theta \leq x_j \leq \theta \qquad (j = 1, \ldots, n) .$$

This can be converted to a simple LP problem in terms of equations in non-negative variables using the transformation $\bar{x}_j = x_j + \theta$ as follows:

(3) Min θ

$$A\bar{x} - (Ae)\theta = b$$

$$\bar{x}_j - 2\theta + s_j = 0 \qquad (j = 1, \ldots, n)$$

$$\bar{x},\ s,\ \theta \geq 0$$

where e is a vector of ones. This is the form adopted by Cadzow [3] and others and has $(m+n)$ constraints in the $(m+n+1)$ variables, $\bar{x}$, s and θ. The number of variables and constraints is discouraging, but it is worth noting that Schrage [5] has devised a "variable upper bound" algorithm capable of handling implicitly constraints such as $\bar{x}_j \leq 2\theta$, which appear in (3).

An alternative transformation of the problem is to note that since $\theta > 0$ (we assume $b \neq 0$) we may write

$$y_j = x_j/\theta = x_j\psi$$

where $\psi = 1/\theta$ and pose the problem in the form:

(4) Max ψ

$$Ay = b\psi$$

$$-1 \leq y_j \leq 1$$

Now the standard parametric programming problem (see, e.g. [1]) is to solve:

$$(5) \qquad \text{Min } c'x$$

$$Ax = b + b^{*}\psi \qquad 0 \leq \psi \leq \bar{\psi}$$

$$x_j \geq 0 \qquad (j = 1, \ldots, n) \cdot$$

This is normally done by solving (5) as a linear program for $\psi = 0$ and then, treating ψ much like a variable, increasing it (making the necessary basis changes to preserve optimality) until either ψ reaches the upper limit $\bar{\psi}$ in which we are interested, or it reaches a maximum beyond which the problem is infeasible. It is clear that we can treat formulation (4) as a special case of (5) by transforming y_j to $\bar{y}_j = y_j + 1$ and obtaining the constraints:

$$(6) \qquad A\bar{y} = (Ae) + b\psi \qquad 0 \leq \psi < \infty$$

$$0 \leq \bar{y}_j \leq 2 \qquad j = 1, \ldots, n$$

The simple upper bounds on the $\bar{y}_j$ are a trivial complication and we have a parametric programming problem with a zero objective row ($c' = 0$) and no upper bound on ψ. Since θ must have a positive minimum for $b \neq 0$ we must find a maximum value of ψ for which the constraints (6) can be satisfied. The minimum ℓ_∞-norm solution x to (1) can then be recovered.

2. Parametric Bound Algorithm

The parametric right hand side problem (6) involves only m non-simple upper bound constraints, and would be a far more efficient method of solving the problem than formulation (3). Furthermore the facilities for solving in this way are available in any modern mathematical programming system (see e.g. [4]). It is possible however to solve problem (2) directly, without any transformation of variables, by combining the ideas of simple upper bounding and parametric programming into a parametric bound algorithm.

The first (unpublished) parametric bound algorithm seems to have been due to E. M. L. Beale, who applied the idea to modifying the bounds on variables in branch-and-bound algorithms for integer and non-convex programming (see, e.g. [6]). Our problem (2) is particularly simple since we have no L.P. objective function and all the variables will be parametrically bounded above (by θ) and below (by $-\theta$). We will use Beale's type of notation [1] and assume some familiarity with the simplex method for L.P.

Let us choose an initial $m \times m$ basis B from the columns of A (we have assumed A has rank m) and partition A into (B,N) after rearranging columns. We denote the variables x_{j_i} corresponding to the i^{th} column of B by X_i and let $\mathcal{N}$ be the set of indices of columns in N, that is the set of indices corresponding to non-basic or independent variables.

Let

$$\bar{a}_j = B^{-1}a_j \qquad \text{for} \quad j \in \mathcal{N} \tag{7}$$

and define

$$\bar{a}_0 = B^{-1}b \tag{8}$$

The equations (1) may then be written in "solved" form

$$X_i = \bar{a}_{i0} + \sum_{j \in \mathcal{N}} \bar{a}_{ij}(-x_j) \qquad (i = 1, \ldots, m). \tag{9}$$

We will allow the non-basic variables to take values of either $-\theta$, 0 or θ and define corresponding index sets:

$$\begin{aligned} \mathcal{L} &= \{j \in \mathcal{N} \mid x_j \text{ is at } -\theta\} \\ \mathcal{Z} &= \{j \in \mathcal{N} \mid x_j \text{ is at } 0\} \\ \mathcal{U} &= \{j \in \mathcal{N} \mid x_j \text{ is at } \theta\} . \end{aligned} \tag{10}$$

Two slack variables are also defined for each of the pairs of inequalities $-\theta \leq x_j \leq \theta$ such that

$$\begin{aligned} x_j &= w_j - \theta, & w_j &\geq 0 \\ x_j &= y_j + \theta, & y_j &\leq 0 \end{aligned} \tag{11}$$

It is convenient to use the w_j, y_j in the description of the algorithm and to define a new symbol for the non-basic variables:

$$(12) \qquad z_j = \begin{cases} w_j & \text{if } j \in \mathscr{L} \\ x_j & \text{if } j \in \mathscr{Z} \\ y_j & \text{if } j \in \mathscr{U} \end{cases}$$

We may now rewrite the "solved" form (9) obtaining

$$X = B^{-1}[b + \sum_{j \in \mathscr{N}} a_j(-x_j)]$$

$$= B^{-1}[b + \sum_{j \in \mathscr{L}} a_j \theta + \sum_{j \in \mathscr{U}} a_j(-\theta) + \sum_{j \in \mathscr{N}} a_j(-z_j)],$$

and define

$$(13) \qquad \bar{a}_0^* = B^{-1}[\sum_{j \in \mathscr{L}} a_j - \sum_{j \in \mathscr{U}} a_j] .$$

The new expression for the basic variables is then

$$(14) \qquad X_i = \bar{a}_{i0} + \bar{a}_{i0}^* \theta + \sum_{j \in \mathscr{N}} a_{ij}(-z_j) \qquad (i = 1, \ldots, m)$$

where all the non-basic variables z_j are at zero.

The initializing procedure is to find any non-singular basis B and set $\mathscr{Z} = \mathscr{N}$, $\mathscr{L} = \mathscr{U} = \emptyset$. The value of X_i with largest modulus is the initial value θ_0.

A general step of the algorithm is derived as follows: From (14) we require the values of the basic variables X_i to satisfy

$$(15) \qquad -\theta \leq \bar{a}_{i0} + \bar{a}^{*}_{i0}\theta \leq \theta \qquad (i = 1, \ldots, m)$$

i.e.

$$(16) \qquad \bar{a}_{i0} + (\bar{a}^{*}_{i0} + 1)\theta \geq 0 \qquad (i = 1, \ldots, m)$$

$$(17) \qquad \bar{a}_{i0} + (\bar{a}^{*}_{i0} - 1)\theta \leq 0 \qquad (i = 1, \ldots, m)$$

and these relationships are satisfied by the current minimum θ.

Now define

$$(18) \qquad \begin{aligned} R_1 &= \{i \,|\, \bar{a}_{i0} < 0,\ \bar{a}^{*}_{i0} + 1 > 0\} \\ R_2 &= \{i \,|\, \bar{a}_{i0} > 0,\ \bar{a}^{*}_{i0} - 1 < 0\} \ . \end{aligned}$$

It is clear that as θ is decreased (16) may be violated iff $i \in R_1$ and (17) may be violated iff $i \in R_2$.

Let

$$(19) \qquad \theta_1 = \max_{i \in R_1} \frac{-\bar{a}_{i0}}{1 + \bar{a}^{*}_{i0}} = \frac{-\bar{a}_{p_1 0}}{1 + \bar{a}^{*}_{p_1 0}}$$

$$(20) \qquad \theta_2 = \max_{i \in R_2} \frac{\bar{a}_{i0}}{1 - \bar{a}^{*}_{i0}} = \frac{\bar{a}_{p_2 0}}{1 - \bar{a}^{*}_{p_2 0}}$$

then the minimum θ that can be obtained with the current basic solution is

(21) $$\bar{\theta} = \max(\theta_1, \theta_2)$$

If the maximum is attained in (21) for θ_k, let $p = p_k$. Then X_p will leave the basis at its lower bound (for $k = 1$) or its upper bound (for $k = 2$).

Case 1, $\bar{\theta} = \theta_1$: In this case $X_p = X_{p_1}$ will go to its lower bound of $-\bar{\theta}$. If we are going to be able to decrease θ further it is clear from (14) that we must find a non-basic z_j such that $\bar{a}_{pj}(-z_j)$ can be made positive. Hence we must either:

(i) increase a non-basic variable with $\bar{a}_{pj} < 0$ and $j \in \mathscr{L} \cup \mathscr{Z}$,

or (ii) decrease a non-basic variable with $\bar{a}_{pj} > 0$ and $j \in \mathscr{U} \cup \mathscr{Z}$.

If such a variable can be found, say z_q, we will perform a simplex pivot and make z_q basic while X_p enters the set $\mathscr{L}$. To do this we must first make the appropriate substitution $w_p - \theta$ for X_p in (14), which amounts to adding 1 to $\bar{a}^*_{p0}$ before pivoting w_p out of the basis to become a z_j. If for the chosen non-basic variable z_q we have $q \in \mathscr{Z}$ the pivot step is then completely straightforward. If however $q \in \mathscr{L}$ we must remove the variable from this set (i.e. subtract $\bar{a}_q$ from $\bar{a}^*_0$) and then pivot on $\bar{a}_{pq}$. It is easy to see from the simplex pivot rules (see [1], p. 22) that this is equivalent to subtracting 1 from $\bar{a}^*_{p0}$ after pivoting. If $q \in \mathscr{U}$ we likewise remove the variable from this set (i.e. add $\bar{a}_q$ to $\bar{a}^*_0$) and then

pivot on $\bar{a}_{pq}$. We see that this is equivalent to adding 1 to $\bar{a}^*_{p0}$ after pivoting. Note that removing z_q from $\mathscr{L}$ or $\mathscr{U}$ is the same as making a transformation from the w_q or y_q variable to the x_q using (11).

Case 2, $\bar{\theta} = \theta_2$: In this case $X_p = X_{p_2}$ will go to its upper bound of $\bar{\theta}$. If we are going to be able to decrease θ further we must find a z_j such that $\bar{a}_{pj}(-z_j)$ can be made negative. Hence we must either:

(i) increase a non-basic variable with $\bar{a}_{pj} > 0$ and $j \in \mathscr{L} \cup \mathscr{Z}$,

or (ii) decrease a non-basic variable with $\bar{a}_{pj} < 0$ and $j \in \mathscr{U} \cup \mathscr{Z}$.

Again if such a z_q can be found we will make the corresponding x_q basic in place of X_p. Since X_p is going to its upper bound the appropriate substitution for X_p is $y_p + \theta$, which amounts to subtracting 1 from $\bar{a}^*_{p0}$ before pivoting y_p out of the basis to become a z_j. When the simplex pivot on $\bar{a}_{pq}$ is performed we must again make appropriate modifications to $\bar{a}^*_{p0}$ after pivoting if $q \in \mathscr{L}$ or $\mathscr{U}$. These are exactly as in Case 1.

When the simplex pivot is completed we return to do the ratio tests (19), (20) on the new "tableau."

If no element $\bar{a}_{pj}$ of the appropriate sign can be found in the "blocking" row p at any step the algorithm terminates with $\bar{\theta}$ the value of the minimum ℓ_∞-norm solution to (1). The solution is immediately available as:

$$x_j = -\bar{\theta} \qquad j \in \mathscr{L}$$

$$x_j = 0 \qquad j \in \mathscr{F}$$

$$x_j = \bar{\theta} \qquad j \in \mathscr{U}$$

$$x_{j_i} = \bar{a}_{i0} + \bar{a}^*_{i0}\bar{\theta} \qquad \text{for basic variables } X_i .$$

3. Efficiency

Since the parametric bound algorithm works with only a system of the size of the original equations, an extra vector $\bar{a}^*_0$ and indicators on the status of the non-basic variables, it is clear that this method will be far more efficient than the primitive L.P. formulation (3), and experiments confirm this even for small problems. To compare the method with Cadzow's special purpose algorithm [3], we will use his criterion of counting operations, assuming the implementation uses an explicit inverse form of the basis. At each step we must

(i) Compute the ratios θ_1, θ_2 : 2m operations

(ii) Compute a vector $\pi'_p = e'_p B^{-1}$
where B^{-1} is the current basis inverse
and e_p is the p^{th} unit vector : m^2 operations

(iii) Compute the n-m inner products
$\bar{a}_{pj} = \pi'_p a_j$: m(n-m) operations

(iv) Update B^{-1} : $2m^2$ operations

(v) Update $\bar{a}_0$ and $\bar{a}^*_0$: 4m operations

This gives a total of $mn + 2m^2 + 6m$ operations. Cadzow gave a figure of $3mn + 3m^2 + n - 2m$ operations for each step of his own algorithm and gives evidence of convergence in m-1 steps. Unless the parametric algorithm requires rather more than $3m$ iterations it would seem to be the more efficient by this criterion. Computational experience so far suggests that the number of iterations required is much smaller (usually of order m or a little less).

The algorithm described in Section 2 leaves two choices ambiguous. The starting basis B is unspecified, but might be chosen to give preference to columns with small norm in the hope of obtaining a starting solution with small initial θ. Any starting basis will however serve the purpose. The choice of $\bar{a}_{pq}$ is also left ambiguous except that it must be non-zero (or at least outside some zero tolerance) and of appropriate sign. On numerical grounds it would be desirable to have $|\bar{a}_{pq}|$ as large as possible and we are free to do so since we are not restricted by having to keep an objective function optimal, as in ordinary parametric programming. Fortunately, this modification also improves convergence (see the experimental results of the next section) since from (14) a large $|\bar{a}_{pq}|$ potentially allows a greater reduction of $|X_p|$ for the same change in the non-basic variable z_q.

4. Implementation and Computational Results

The parametric bound algorithm was implemented by making simple modifications of a FORTRAN all-in-core linear programming code using sparse matrix techniques and a product form of inverse. This code did not employ the simple upper bound algorithm, but all that was required was the addition of an indicator to each non-basic column, specifying whether it belonged to $\mathscr{L}$, $\mathscr{Z}$ or $\mathscr{U}$. The initial basis is chosen by simply taking the first m columns and trying to form a product form inverse (more precisely a product form inverse of the LU decomposition) and rejecting any vectors found to be dependent. The matrix is then scanned for vectors independent of those already included. The first version of the code chose the first $\bar{a}_{pj}$ of the appropriate sign for a pivot and the number of iterations required is shown in the column headed "arbitrary $\bar{a}_{pj}$" in Table 1. Not surprisingly this scheme was inadequate for larger problems and the code was changed to pick the largest eligible $|\bar{a}_{pj}|$. The number of iterations using this scheme is given in the column headed "max$|\bar{a}_{pj}|$" in Table 1. Some of the smaller problems were solved by using the "primitive LP" formulation (3) and the number of iterations is given in the column so headed. All but the two smallest examples are linear programming constraint matrices (in the form of equations naturally, not inequalities) and are therefore sparse. Their densities are given in Table 1. It seems probable that this sparsity assists convergence, since a proportion of the $\bar{a}_{pj}$ will be zero and hence ineligible pivots, but the effect is probably not crucial. In any case sparsity is usually a characteristic of large systems of equations.

Experiments not performed are indicated in Table 1 by a dash.

TABLE 1

COMPUTATIONAL RESULTS

Problem	m	n	% density	Iterations		
				max$\lvert \bar{a}_{pj} \rvert$	arbitrary $\bar{a}_{pj}$	primitive L.P.
1	3	5	100	--	4	7
2	4	8	90.6	--	4	8
3	28	51	7.5	--	30	82
4	57	97	8.4	6	37	> 138
5	99	162	5.5	76	> 900	--
6	174	297	6.8	61	--	--
7	272	482	1.5	137	--	--

5. Conclusion

The experiments described in Section 4 indicate that the parametric bound algorithm is a very efficient and direct approach to the minimum ℓ_∞-norm problem not only in operations per step, but in number of iterations required. Furthermore, as a fairly simple variation of the standard parametric linear programming method it is extremely easy to implement in any L.P. system containing such facilities, and thus has available to it all the power of such a system.

REFERENCES

[1] E. M. L. Beale, Mathematical Programming in Practice, Pitmans, London and Wiley, New York (1968).

[2] E. M. L. Beale, "Advanced Algorithmic Features for General Mathematical Programming Systems" in Integer and Non-Linear Programming (Ed., J. Abadie) North Holland, Amsterdam (1970), pp. 119-137.

[3] J. A. Cadzow, "An Efficient Algorithmic Procedure for Obtaining a Minimum ℓ_∞-Norm Solution to a System of Consistent Linear Equations." SIAM J. Num. Anal. 11 (1974), pp. 1151-1165.

[4] IBM Corporation, "Mathematical Programming System-Extended (MPSX) and Generalized Upper Bounding (GUB)." Program Description. Document No. SH20-0968-1 (1972).

[5] L. Schrage, "Implicit Representation of Variable Upper Bounds in Linear Programming." Manuscript, UCLA, October 1973.

[6] J. A. Tomlin, "Branch and Bound Methods for Integer and Non-Convex Programming" in Integer and Non-Linear Programming (Ed., J. Abadie) North Holland, Amsterdam (1970), pp. 437-450.

[7] H. C. Torng, "Optimization of Discrete Control Systems Through Linear Programming," J. Franklin Inst. 278 (1964), pp. 28-44.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SOL 75-12	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A PARAMETRIC BOUNDING METHOD FOR FINDING A MINIMUM ℓ_∞-NORM SOLUTION TO A SYSTEM OF EQUATIONS		5. TYPE OF REPORT & PERIOD COVERED Technical Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) J. A. TOMLIN		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0865 N00014-75-C-0267 DAHC04 74 C 0034
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Operations Research Stanford University Stanford, CA 94305		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR 047-143 NR 047-064 P-12215-M
11. CONTROLLING OFFICE NAME AND ADDRESS Operations Research Program Code 434 Office of Naval Research Arlington, Virginia 22217 Mathematics Division U.S. Army Research Office Box CM, Duke Station Durham, North Carolina 27706		12. REPORT DATE May 1975 13. NUMBER OF PAGES 16 15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and sale; its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

LINEAR PROGRAMMING
PARAMETRIC PROGRAMMING
DISCRETE CONTROL SYSTEMS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper presents a method for finding the minimum ℓ_∞-norm solution to a set of consistent linear equations using a form of parametric linear programming. In this application the upper and lower bounds of all the variables are parametrized, and we work with only the original variables and constraints. Computational results indicate that the method is superior to both a primitive linear programming approach to the problem and to other, more specialized methods, which have been suggested.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)