Generalizing the LINPACK Condition Estimator

A. K. Cline

A. R. Conn*

C. Van Loan**

June 1981

TR 81-462

Department of Computer Science
Cornell University
Ithaca, New York 14853

*This work was supported, in part, by NSERC Grant A8639.
**This work was supported, in part, by NSF Grant MCS 80-04106.

GENERALIZING THE LINPACK CONDITION ESTIMATOR

Alan K. Cline
Department of Computer Science
University of Texas
Austin, Texas 78712

Andrew R. Conn
Department of Computer Science
University of Waterloo
Waterloo, Ontario N2L 3G1
Canada

Charles F. Van Loan
Department of Computer Science
Cornell University
Ithaca, New York 14853

Abstract

Two generalizations of the Cline-Moler-Stewart-Wilkinson "LINPACK" condition estimator are described. One generalization combines the LINPACK notion of "look-ahead" with a new feature called "look-behind" that results in a more flexibly chosen right-hand side. The other generalization is a "divide-and-conquer" scheme that involves estimating the condition of certain principal submatrices whose dimension repeatedly doubles. Both generalizations require the maximization of simple objective functions. When seeking an L_1 condition estimate, these functions are convex while in the L_2 case they are quadratic. All the algorithms considered appear to be at least as reliable as the LINPACK estimator and are equally efficient.

1. Background

Suppose the n-by-n nonsingular linear system $Ax = b$ is solved using a "stable" matrix factorization method such as Gaussian elimination with partial pivoting. If t-digit, base b floating point arithmetic is used, then it is generally the case that the relative error in the computed solution $\hat{x}$ satisfies

$$(1) \qquad \frac{\| \hat{x} - x \|_p}{\| x \|_p} \cong b^{-t} k_p(A).$$

Here, $\| x \|_p = (|x_1|^p + \ldots + |x_n|^p)^{1/p}$ and $k_p(A)$ is the p-norm condition of A defined by

$$k_p(A) = \max_{z \neq 0} \frac{\| Az \|_p}{\| z \|_p} \Big/ \min_{z \neq 0} \frac{\| Az \|_p}{\| z \|_p} \equiv \| A \|_p \| A^{-1} \|_p \ .$$

Heuristic (1) implies that an estimate $\hat{k}_p$ to $k_p(A)$ can be useful when assessing the quality of $\hat{x}$.

Among other things, the attractiveness of a condition estimator depends upon its reliability and how expensive it is to compute. With respect to reliability, we adopt the convention that $\hat{k}_p$ is a reliable estimator if

$$(2) \qquad c_1 \, k_p(A) \leqslant \hat{k}_p \leqslant c_2 \, k_p(A)$$

for "reasonable" constants c_1 and c_2 that are independent of A.

Consider, for example, the estimator $\hat{k}_2 = \hat{\sigma}_1 / \hat{\sigma}_n$ where $\sigma_1 \geqslant \ldots \geqslant \sigma_n \geqslant 0$ are the singular values of A computed by either the LINPACK [2] or EISPACK [7] singular value decomposition (SVD) subroutines. This estimate is provably reliable in that it can be rigorously shown that the constants in (2) are each of the form $1 + O(b^{-t})$. Unfortunately, computing the SVD requires about 15 times as many flops as Gaussian elimination and so this is a rather expensive method for assessing $\hat{x}$. (A "flop" is floating point multiplicative operation.) Condition estimators that require only $O(n^2)$ flops once we have computed a "cheap" factorization such as PA = LU (via Gaussian elimination with partial pivoting) or AP = QR (via Householder triangularization with column pivoting) are therefore of interest.

Note that with such a factorization available we can readily compute $\| A \|_p \, \| \hat{X} \|_p \cong k_p(A)$ where $p = 1$ or ∞ and $\hat{X}$ is the computed solution to $AX = I$. Although estimators of this type are provably reliable, they are inefficient because they require $O(n^3)$ flops. The challenge is thus to reliably estimate the condition in $O(n^2)$ flops assuming that A has already been factored.

Forsythe and Moler [3] propose an interesting $O(n^2)$ estimator based on iterative improvement and the assumption that $\hat{x}$ has been computed via PA = LU . In particular, they set $\hat{k}_\infty = b^t \, \| \hat{z} \|_\infty / \| \hat{x} \|_\infty$ where z is computed by solving $Lw = Pr$ and $Uz = w$ where the residual $r = b - A\hat{x}$ is calculated in double precision. Although this estimator is efficient, its attractiveness is limited (a) because of portability problems associated with the double precision calculation of r and (b) because an extra n-by-n array is required. Its reliability is unproven but it appears to be a successful estimation technique.

Karasalo [5] describes an efficient 2-norm condition estimator that is based on properties of the triangular matrix R that is computed via Householder triangularization with column pivoting. However, the estimator is not provably reliable to the extent that the constant c_1 in (2) must be of order 4^{-n}.

2. The LINPACK Approach

Another approach to the condition estimation problem is taken by Cline, Moler, Stewart, and Wilkinson [1] and is implemented in LINPACK. It amounts to inverse iteration with a special technique for computing the starting vector. The method assumes that A has been factored and proceeds as follows:

Step 1. Choose d such that the solution to $A^T w = d$ is large in norm relative to d.

Step 2. Solve $Az = w$.

Step 3. Set $\hat{k}_1 = \| A \|_1 \; \| z \|_1 / \| w \|_1$

Note that since $\| A^{-1} \|_1 \geq \| z \|_1 / \| w \|_1$ we have

$$\frac{\| z \|_1}{\| A^{-1} \|_1 \| w \|_1} k_1(A) = \hat{k}_1 \leq k_1(A)$$

Hence, from the standpoint of reliability, it is desirable that the quotient $\| z \|_1 / \| w \|_1$ be as close to $\| A^{-1} \|_1$ as possible. That Step 1 encourages this can be seen via a brief 2-norm argument using the SVD. Let

$$A = U \operatorname{diag}(\sigma_i) V^T \qquad U^T U = I, \; V^T V = I, \; \sigma_1 \geq \cdots \geq \sigma_n \geq 0$$

be the SVD of A with $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_n]$. If

$$d = \sum_{i=1}^{n} a_i v_i$$

then

(3) $$w = \sum_{i=1}^{n} \frac{a_i}{\sigma_i} u_i \qquad \text{and} \qquad z = \sum_{i=1}^{n} \frac{a_i}{\sigma_i^2} v_i \;.$$

A calculation shows that

$$\left(\frac{\| z \|_2}{\| w \|_2}\right)^2 \geqslant \frac{a_n^2}{\sigma_n^2 \, \| d \|_2^2} \; \frac{\sum_{i=1}^{n} a_i^2}{\sum_{i=1}^{n} \left(\frac{\sigma_n}{\sigma_i}\right)^2 a_i^2}$$

i.e.,

$$\frac{\| z \|_2}{\| w \|_2} \geqslant \| A^{-1} \|_2 \; \cos(v_n, d) \qquad , \qquad \cos(v_n, d) = \frac{\| a_n \|}{\| d \|_2} \;.$$

Thus, it is desirable that $\cos(v_n,d)$ be near unity. As (3) suggests, striving for a large w in Step 1 tends to produce a vector d that has a significant component in the direction of v_n.

To motivate the LINPACK method for carrying out Step 1, assume that T is an n-by-n lower triangular matrix and consider the problem of choosing d such that the solution to $Ty = d$ has a large norm. Note that y is given by

p_k := 0 $\qquad (k = 1,\dots,n)$
For $k = 1,\dots,n$
$\quad y_k$:= $(d_k - p_k) / t_{kk}$
$\quad p_i$:= $p_i + t_{ik} y_k \qquad (i = k+1,\dots,n)$

Hence, it is desirable that d_k be chosen so that both y_k and the running sums $p_{k+1},\ldots,p_n$ are as large as possible. Towards this end we can set $d_k = a$ where $a \in \{-1,+1\}$ maximizes

$$\phi_k(a) = |y_k(a)| + \sum_{i=k+1}^{n} w_i \, |p_i + t_{ik} y_k(a)| .$$

Here, $y_k(a) = (a - p_k)/t_{kk}$ and the w_i are nonnegative weights. In LINPACK, these weights are all set to 1. Another possibility mentioned in [1] is to set $w_i = 1/\, t_{ii}$.

If A is square and $PA = LU$, then the LINPACK estimator determines the vector d in Step 1 by applying the above scheme with $T = U^T$. Note that Steps 1-3 require $O(n^2)$ flops and so the method is efficient. However, its success depends on an additional heuristic, namely, that by striving for a large norm solution in $U^Ty = d$, we obtain a large norm solution to $A^Tw = d$. Experimental evidence suggests that this is frequently the case but we will comment more fully on the method's reliability in Section 6.

3. Estimators with "Look Behind"

In this and the next section we assume that $A = T$ is lower triangular and we consider various alternatives to the LINPACK method for producing a large norm solution to $Ty = d$. Our first alternative incorporates the notion of "look behind" and we begin by developing a 2-norm condition estimator that has this feature. For the sake of clarity, assume $n=6$ and that d_1, d_2, and d_3 are known and satisfy $d_1^2 + d_2^2 + d_3^2 = 1$. Also assume that we have solved the system

$$(4) \qquad \begin{bmatrix} t_{11} & 0 & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & t_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix},$$

and have computed the associated "look-ahead" values:

$$(5) \qquad \begin{aligned} p_4 &= t_{41}y_1 + t_{42}y_2 + t_{43}y_3 \\ p_5 &= t_{51}y_1 + t_{52}y_2 + t_{53}y_3 \\ p_6 &= t_{61}y_1 + t_{62}y_2 + t_{63}y_3. \end{aligned}$$

We now determine $c = \cos(a)$ and $s = \sin(a)$ such that if

$$\begin{bmatrix} t_{11} & 0 & 0 & 0 \\ t_{21} & t_{22} & 0 & 0 \\ t_{31} & t_{32} & t_{33} & 0 \\ t_{41} & t_{42} & t_{43} & t_{44} \end{bmatrix} \begin{bmatrix} y_1' \\ y_2' \\ y_3' \\ y_4' \end{bmatrix} = \begin{bmatrix} s\ d_1 \\ s\ d_2 \\ s\ d_3 \\ c \end{bmatrix}$$

then $\sum_{i=1}^{4} (y_i')^2 + \sum_{i=5}^{6} (p_i')^2$ is maximized where $p_i' = sp_i + t_{i4}y_4'$, $i=5,6$.

(The p_i' are the updates of the p_i.) Notice that by revising the right hand side in this fashion the solution of the "enlarged" system is easily computed:

$$y_i' = sy_i \quad (i = 1,2,3)$$

$$y_4' = (c - sp_4)/t_{44}.$$

Also observe that in the course of doing these calculations d_1, d_2, and d_3 are revised downwards by a factor of s and that the new right hand side has unit 2-norm.

In general, at the k-th step when d_k is calculated, we "look behind" and consider the revision of $d_1,..,d_{k-1}$ and we "look ahead" to anticipate the effects on $p_{k+1},...,p_n$. Overall we have

Algorithm 1

$p_k := 0 \quad (k = 1,...,n)$

For $k = 1,...,n$

1. Determine $a \in [0,2\pi]$ such that if $c = \cos(a)$, $s = \sin(a)$, and $y_k(a) = (c - sp_k)/t_{kk}$ then

$$\phi_k(a) = s^2 \sum_{i=1}^{k-1} y_i^2 + y_k(a)^2 + \sum_{i=k+1}^{n} w_i^2(sp_i + t_{ik}y_k(a))^2$$

   is maximized where $w_1,...,w_n$ are fixed nonnegative weights.

2. $c := \cos(a)$; $s := \sin(a)$; $d_k := c$; $y_k := (d_k - sp_k)/t_{kk}$;

   $d_i := s\, d_i \quad (i = 1,..,k-1)$

   $y_i := s\, y_i \quad (i = 1,..,k-1)$

   $p_i := sp_i + t_{ik}y_k \quad (i = k+1,...,n)$

The parameter a is easily determined. From the equation $\phi'_k(a) = 0$ we obtain the relation

$$(6) \qquad \text{beta} * c * s = \text{alfa} * (c^2 - s^2)$$

where

$$\text{beta} = (y^T y + p^T D^2 p) t_{kk}^2 + (p_k^2 - 1)(1 + t^T D^2 t) - 2 p_k t_{kk} p^T D^2 t$$

$$\text{alfa} = p_k (1 + t^T D^2 t) - t_{kk} \, p^T D^2 t$$

$$t^T = (t_{k+1,k}, \ldots, t_{n,k})$$
$$p^T = (p_{k+1}, \ . \ . \ . \ , p_n)$$
$$y^T = (y_1, \ . \ . \ . \ , y_{k-1})$$

and

$$D = \text{diag}(w_{k+1}, \ . \ . \ . \ , w_n).$$

The two possible sine-cosine pairs that satisfy (6) can be calculated as follows:

$$r := \text{beta}/(2*\text{alfa});$$
$$\mu_1 := r + \text{sqrt}(1 + r^2); \quad s_1 := 1/\text{sqrt}(1 + \mu_1^2); \quad c_1 := s_1 \, \mu_1$$
$$\mu_2 := r - \text{sqrt}(1 + r^2); \quad s_2 := 1/\text{sqrt}(1 + \mu_2^2); \quad c_2 := s_2 \, \mu_2$$

The pair that maximizes $\phi_k(a)$ can be determined via substitution.

Algorithm 1 requires approximately $5n^2$ flops and is readily seen to produce the estimate

$$(7) \qquad (\sigma_n, v_n, u_n) \cong (\hat{\sigma}_n, \hat{v}_n, \hat{u}_n) \equiv \left(\frac{1}{\| y \|_2} , \ d \ , \ \frac{y}{\| y \|_2} \right)$$

where σ_n is the n-th singular value of T and v_n and u_n are the associated right and left singular vectors. On the other hand, if a is chosen at each stage so as to minimize $\phi_k(a)$, then an estimate of the largest singular value and its singular vectors result:

$$(8) \qquad (\sigma_1, v_1, u_1) \cong (\hat{\sigma}_1, \hat{v}_1, \hat{u}_1) = \left(\frac{1}{\| y \|_2}, d, \frac{y}{\| y \|_2} \right)$$

Of course, (7) and (8) combine to give $\hat{k}_2 = \hat{\sigma}_1 / \hat{\sigma}_n$.

An L_1 "look behind" condition estimator can also be devised. To illustrate, suppose $n = 6$, $k = 3$, and that equations (4) and (5) hold with $|d_1| + |d_2| + |d_3| = 1$. We then seek $\lambda \in [0,1]$ such that if

$$\begin{bmatrix} t_{11} & 0 & 0 & 0 \\ t_{21} & t_{22} & 0 & 0 \\ t_{31} & t_{32} & t_{33} & 0 \\ t_{41} & t_{42} & t_{43} & t_{44} \end{bmatrix} \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix} = \begin{bmatrix} \lambda d_1 \\ \lambda d_2 \\ \lambda d_3 \\ 1 - \lambda \end{bmatrix}$$

then $\sum_{i=1}^{4} |y'_i| + \sum_{i=5}^{6} |p'_i|$ is maximized where $p'_i = p_i + t_{i4} y'_4$, $i=5,6$. Since the function to be maximized is convex, it suffices merely to check its value at $\lambda = 0$ and $\lambda = 1$. In general we have:

Algorithm 2

$p_k := 0 \quad (k = 1,\dots,n)$

For $k = 1$ to n

1. Determine $\lambda \in \{0,1\}$ such that if $y_k(\lambda) = [(1-\lambda) - p_k]/t_{kk}$ then

$$\phi_k(\lambda) = \sum_{i=1}^{k-1} |y_i| + |y_k(\lambda)| + \sum_{i=k+1}^{n} w_i \, |\lambda p_i + t_{ik} y_k(\lambda)|$$

is maximized where the w_i are fixed nonnegative weights.

2. $d_k := 1 - \lambda; \quad y_k := (d_k - \lambda p_k)/t_{kk};$

$d_i := \lambda d_i \quad (i=1,\dots,k-1)$

$y_i := \lambda y_i \quad (i=1,\dots,k-1)$

$p_i := \lambda p_i + t_{ik} y_k \quad (i=k+1,\dots,n)$

With y calculated in this fashion, we obtain the estimate $\hat{k}_1 = \|T\|_1 \|y\|_1$.

Note that the final right hand side d will be some column of the identity which implies that y is a column of T^{-1}. We also remark that Algorithm 2 is considerably more efficient than Algorithm 1, especially since the parameter λ is either zero or one.

4. A Divide and Conquer Estimator

Suppose $T_{11} \in R^{p \times p}$ and $T_{22} \in R^{q \times q}$ are lower triangular and that we have determined y_1, y_2, d_1, and d_2 so

$$T_{11} y_1 = d_1 \qquad \| d_1 \|_2 = 1$$

$$T_{22} y_2 = d_2 \qquad \| d_2 \|_2 = 1$$

Consider the problem of choosing $c = \cos(a)$ and $s = \sin(a)$ such that if

$$\begin{bmatrix} T_{11} & 0 \\ T_{21} & T_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} c\, d_1 \\ s\, d_2 \end{bmatrix}$$

then

$$\phi(a) = \| z_1 \|_2^2 + \| z_2 \|_2^2$$

is maximized. Define w by $T_{22} w = T_{21} y_1$. A calculation shows that $z_1 = c\, y_1$ and $z_2 = s y_2 - c w$ and thus,

$$(9) \qquad \phi(a) = c^2 [\| y_1 \|_2^2 + \| w \|_2^2] - 2 s c y_2^T w + s^2 \| y_2 \|_2^2 .$$

By manipulating the equation $\phi'(a) = 0$, we obtain the following formulae for the sines and cosines:

$$\text{beta} := y_2^T y_2 - y_1^T y_1 - w^T w$$
$$\text{alfa} := y_2^T w$$
$$r := \text{beta}/(2*\text{alfa})$$
$$\mu_1 := r + \text{sqrt}(1 + r^2); \; s_1 := 1/\text{sqrt}(1 + \mu_1^2); \; c_1 := \mu_1 s_1$$
$$\mu_2 := r - \text{sqrt}(1 + r^2); \; s_2 := 1/\text{sqrt}(1 + \mu_2^2); \; c_2 := \mu_2 s_2$$

Again, the sine-cosine pair that maximizes ϕ can be determined by substitution.

This computation forms the heart of a divide-and-conquer algorithm that can be used to produce a large norm solution to $Ty = d$. Consider the case n=8. We begin by solving the eight 1-by-1 systems $(t_{ii})y_i = 1$. These linear systems are then paired and combined in the above fashion to produce four 2-by-2 systems that involve the matrices

$$\begin{bmatrix} t_{ii} & 0 \\ t_{ji} & t_{jj} \end{bmatrix} \qquad i = 1,3,5,7 \; ; \quad j = i+1$$

These systems are in turn paired and combined, all the while choosing the sines and cosines to encourage growth. Finally, the two 4-by-4 systems are synthesized to render a final d ($\|d\|_2 = 1$) and y (hopefully large in norm) such that $Ty = d$.

In the general case there are several ways to handle the pairing of the systems in the event that the dimension of T is not an exact power of 2 . Our approach is as follows. Suppose at some stage we have solved k linear systems $S_1,\ldots,S_k$ associated with k principal submatrices that are ordered along the diagonal of T. Our task is to pair and combine these linear systems together according to the scheme described above. Write $k = 2p + q$ where q is either zero or one. For $i = 1,\ldots,p$ we combine S_{2i-1} and S_{2i} to produce S'_i. If $q=0$ then we move on to the next stage with the systems $S'_1,\ldots,S'_p$. Otherwise, $q = 1$ and we combine S'_p with S_k to produce S''_p and proceed to the next stage with the systems $S'_1,\ldots,S'_{p-1}$, S''_p.

We emerge from the overall procedure with an estimate of the form (7). If ϕ is minimized at each step, then the estimate (8) for the largest singular value results.

The divide and conquer scheme requires a few n^2 flops and can obviously be adapted to render an L_1 condition estimator.

5. Test Results

The above condition estimators have been tested on numerous examples. In the L_2 case, we examined how well

E1 : Divide and Conquer

E2 : Look-Behind (Algorithm 1) with weights $w_i = 1/t_{ii}$

E3 : Look-Behind (Algorithm 1) with weights $w_i = 1$.

could estimate the largest and smallest singular values of a given lower triangular matrix T.

Test 1.

- The lower triangular elements of T were randomly selected from [-1,+1].
- 1000 examples were run; 100 each for n = 5,10,15,20,25,30,35,40,45,50 .
- The following table reports on the distribution of the "success" measures $q_n = \sigma_n/\hat{\sigma}_n$ and $q_1 = \hat{\sigma}_1/\sigma_1$

Greater Than	But Less Than	E1		E2		E3	
		q_n	q_1	q_n	q_1	q_n	q_1
.9	1.0	65.1%	1.1%	56.8%	1.0%	62.6%	1.7%
.8	.9	12.4%	2.4%	11.1%	1.2%	11.6%	1.8%
.7	.8	6.1%	1.9%	7.9%	1.7%	6.8%	2.6%
.6	.7	4.7%	3.9%	4.8%	6.9%	3.5%	4.9%
.5	.6	4.0%	4.7%	4.2%	20.5%	4.4%	10.6%
.4	.5	2.9%	8.6%	4.7%	43.5%	2.8%	38.8%
.3	.4	2.1%	20.2%	3.4%	23.2%	3.2%	38.3%
.2	.3	1.2%	36.4%	2.6%	1.7%	1.8%	1.8%
.1	.2	1.2%	18.8%	3.2%	.1%	2.5%	.0%
.0	.1	.2%	2.0%	3.2%	.0%	.3%	.0%

Comments

- We discovered no correlation between the quality of the estimates and either the dimension of the matrix or the condition of the matrix.
- The choice of the w_i in the Look-Behind schemes does not seem too important.
- We have no explanation why the estimates of σ_1 are consistently inferior to those for σ_n.
- In no instance was either q_1 or q_n less than .05 .

Test 2

- T generated by computing the QR-with-column-pivoting factorization of a square matrix A whose entries are randomly selected from $[-1,+1]$. More precisely, the factorization $AP = QT$ was computed where T is lower triangular and P is chosen to maximize t_{kk} $(k=n,\ldots,1)$ at each step. (It is obviously desirable to have the small elements in the northwest corner of T.)
- 1000 examples were run; 100 each for n=5,10,15,20,25,30,35,40,45,50.
- The following table reports on the distribution of $q_n = \sigma_n/\hat{\sigma}_n$.

Greater Than	But Less Than	E1 q_n	E2 q_n	E3 q_n
.9	1.0	97.5%	98.9%	97.4%
.8	.9	1.7%	.5%	1.4%
.7	.8	.7%	.5%	.9%
.6	.7	.0%	.0%	.2%
.5	.6	.1%	.1%	.1%
.0	.5	.0	.0	.0

Comments

- By virtue of the column pivoting, $t_{kk}^2 \geq \sum_{i=1}^{k} t_{ij}^2$ for all $1 \leq j < k \leq n$.

- The estimates of σ_1 were of the same quality as in Test 1 and so were unnecessary to report.

- In the majority of cases, $q_n > .99$

We remark that in both of the above tests, $\hat{k}_2 = \hat{\sigma}_1/\hat{\sigma}_n$ was, in the vast majority of cases, within a factor of 10 of $k_2(T)$. Note that if we performed one step of inverse iteration, an even higher quality estimate would result.

The L_1 look-behind technique (Algorithm 2 with $w_i = 1$) was also tested and found to be comparably reliable:

Test 3.

- The lower triangular elements of T were randomly selected from $[-1,+1]$.
- 250 examples were run, 5 each for $n = 1,2,\ldots,50$.
- The following table reports on the approximate distribution of $\hat{k}_1/k_1(T)$

Greater Than	But Less Than	$\dfrac{\hat{k}_1}{k_1(T)}$
.99	1.00	78%
.50	.99	6%
.10	.50	12%
.05	.10	4%

6. Conclusions

It is important to interpret the above experimental results correctly. To begin with, they are just that--experimental results. They do not "prove" anything. However, they do suggest that our methods are at least as reliable as the LINPACK estimator which almost always produces L_1 estimates that are within a factor of 10 of the true condition. Should we therefore argue for the inclusion of one of our methods in LINPACK, especially since Cline [8] has produced an example upon which the LINPACK estimator fails?

This question focuses attention on the difficult problem of assessing condition estimation algorithms. Should one's enthusiasm for a 99% reliable method be diminished because of the existence of counter examples? If we believed this then we would not have presented the divide and conquer technique because it is easy to construct examples upon which it gives arbitrariliy poor estimates! (The counter-example matrix is in fact the same 4-by-4 matrix that appears in [1].) No, we must not at this time argue about whose condition estimator is "better." Instead, we must work to produce an efficient <u>provably</u> <u>reliable</u> technique. We are personally excited by our approaches because the experimental results, particularly Test 2, suggest that some rigorous result may be possible for the case when T is obtained via the QR factorization with pivoting.

Acknowledgements

We are grateful to Cleve Moler, David Gay, and Don Heller for sharing their thoughts on the problems discussed in this paper.

References

[1] A.K.Cline, C.B.Moler, G.W.Stewart, and J.H.Wilkinson, "An Estimate of the Condition Number of a Matrix," SIAM J.Numer.Anal., 16(1979),368-375.

[2] J.J.Dongarra, C.B.Moler, J.R.Bunch, and G.W.Stewart, LINPACK User's Guide, SIAM, Philadelphia, 1979.

[3] G.E.Forsythe and C.B.Moler, Computer Solution of Linear Algebraic Equations, Prentice-Hall, Englewood Cliffs, 1967.

[4] C.B.Moler, "Iterative Refinement in Floating Point," JACM, 14 (1967),316-327.

[5] I.Karasalo, "A Criterion for Trunacation of the QR Decomposition Algorithm for the Singular Linear Least Squares Problem," BIT 14 (1974),116-166.

[6] D.P.O'Leary, "Esimating Matrix Condition Numbers," SISSC 1 (1980) 205-209.

[7] B.Smith et al, EISPACK Guide, Springer-Verlag, New York, 1974.

[8] A.K.Cline, " A Set of Counter Examples to the LINPACK Condition Estimator," Manuscript, Department of Computer Science, University of Texas, Austin TX, 1981.