REMOVAL OF ILL-CONDITIONING FOR MATRICES*

By KURT EISEMANN
(*IBM Corporation, New York*)

Objective. The most common problem in numerical computation is the solution of simultaneous linear equations. The most frequent obstacle is ill-conditioning of the matrix of coefficients, resulting in substantial loss of accuracy. The present paper describes a method of overcoming this difficulty as soon as encountered—with a minimum of computational effort, preservation of reasonable accuracy, and emergence of a well-conditioned system.

The present method applies only to cases in which the equations describe a well-behaved system but are poorly specified in the sense that one or a few of the given equations are almost a linear combination of other equations. The method does *not* apply to cases like the Hilbert matrices, in which *every* row is nearly a linear combination of other rows.

To avoid encumbrance by irrelevant details and to fix ideas, we shall illustrate by a matrix A of order $n = 5$ throughout.

The mathematical key. With high-speed computers, the preferred method of solution is some variant of the Gauss elimination procedure. After k stages, A has been reduced to a form $R^{(k)}$ which, for $k = 2$, is illustrated by

$$R^{(2)} = \begin{bmatrix} 1 & r_{12} & r_{13} & r_{14} & r_{15} \\ & 1 & r_{23} & r_{24} & r_{25} \\ & & s_{33} & s_{34} & s_{35} \\ & & s_{43} & s_{44} & s_{45} \\ & & s_{53} & s_{54} & s_{55} \end{bmatrix}.$$

In general we can write

$$R^{(k)} = \begin{pmatrix} R_{11} & R_{12} \\ 0 & S \end{pmatrix}, \quad \text{where} \quad \begin{cases} R_{11} = \text{upper triangular } k \times k, \\ S = S^{(k)} = (n-k) \times (n-k). \end{cases}$$

Retracing backwards the steps of elimination, matrix $R^{(1)}$ is retrieved from $R^{(2)}$ as

$$R^{(1)} = T^{(2)}R^{(2)}, \quad \text{where} \quad T^{(2)} = \begin{bmatrix} 1 & & & & \\ 0 & l_{22} & & & \\ 0 & l_{32} & 1 & & \\ 0 & l_{42} & 0 & 1 & \\ 0 & l_{52} & 0 & 0 & 1 \end{bmatrix},$$

the elements l_{i2} of column 2 being precisely the elements $s_{i2}^{(1)}$ in the leftmost column of

*Received Aug. 10, 1956.

$S^{(1)}$. Continuing in this manner, matrix A is expressed in the form

$$A \equiv R^{(0)} = T^{(1)}T^{(2)} \cdots T^{(k)}R^{(k)} = L^{(k)}R^{(k)},$$

where $L^{(k)} = T^{(1)}T^{(2)} \cdots T^{(k)}$ is illustrated, for $k = 2$, by

$$L^{(2)} = T^{(1)}T^{(2)} = \begin{bmatrix} l_{11} & & & & \\ l_{21} & 1 & & & \\ l_{31} & 0 & 1 & & \\ l_{41} & 0 & 0 & 1 & \\ l_{51} & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & & & & \\ 0 & l_{22} & & & \\ 0 & l_{32} & 1 & & \\ 0 & l_{42} & 0 & 1 & \\ 0 & l_{52} & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} l_{11} & & & & \\ l_{21} & l_{22} & & & \\ l_{31} & l_{32} & 1 & & \\ l_{41} & l_{42} & 0 & 1 & \\ l_{51} & l_{52} & 0 & 0 & 1 \end{bmatrix}.$$

Similarly, in general,

$$L^{(k)} = T^{(1)}T^{(2)} \cdots T^{(k)} = \begin{pmatrix} L_{11} & 0 \\ L_{21} & I \end{pmatrix}, \quad \text{where} \quad \begin{aligned} L_{11} &= \text{lower triangular } k \times k \\ I &= (n-k) \times (n-k) \end{aligned}$$

As seen above, the elements l_{ik} are furnished directly by the elimination procedure: the leftmost column of $S^{(k)}$ always becomes, at the next stage, the new column $k+1$ of $L^{(k+1)}$.

It is noteworthy that matrices $L^{(k)}$, $R^{(k)}$ and the factorization $A = L^{(k)}R^{(k)}$ represent a generalization of the usual formula $A = LR \equiv L^{(n)}R^{(n)}$ in which L and R are postulated to be triangular.

We shall subsequently also need $P^{(k)}$ = inverse of $L^{(k)}$, which can be written

$$P^{(k)} = \begin{pmatrix} P_{11} & 0 \\ P_{21} & I \end{pmatrix}, \quad \text{where} \quad \begin{aligned} P_{11} &= \text{lower triangular } k \times k, \\ I &= (n-k) \times (n-k). \end{aligned}$$

In the sequel we shall omit superscript k, it being understood that all matrices refer to stage k. Instead, superscripts will henceforth denote row numbers.

The source of trouble. Now ill-conditioning reveals itself by the fact that at some stage in the elimination procedure an entire row of S takes on very small values.

Consider the case where all elements of row S^i are very small. Elimination will then lead, at the present or some later stage, to division of this row by one of its very small current elements in order to provide a coefficient of value 1. This is equivalent to multiplication of row i by a very large number. Accumulated round-off errors are thereby greatly magnified. It is the high *relative* errors infecting this row that introduce the sudden substantial drop in accuracy, subsequently propagating to the entire solution vector. The situation could be saved if it were possible to obtain row S^i, which is the result of many computational steps, free of the high accumulated round-off errors. We shall show how to accomplish this with a comparatively small amount of labor.

The remedy. S^i is equivalent to the result of subtracting a linear combination of rows $A^1 \cdots A^k$ from A^i. We would first like to determine approximately the constants of this linear combination, indicating in what way S^i is obtainable directly from the given original matrix A.

From $A = LR$ (stage k understood) we obtain $R = L^{-1}A = PA$. We can therefore obtain S^i $(i > k)$ from

$$S^i \sim R^i = P^iA = (P^i_{21}, I^i)A = p_{i1}A^1 + \cdots + p_{ik}A^k + A^i.$$

Since the elements of the left-hand side are very small, this relation indicates that $p_{i1} \cdots p_{ik}$ are the negatives of the constants of linear combination of rows $A^1 \cdots A^k$ approximately furnishing A^i.

We therefore need the elements of P^i for $i > k$, i.e. of P^i_{21}. They are found by considering

$$PL = I, \qquad \text{i.e.,} \qquad \begin{pmatrix} P_{11} & 0 \\ P_{21} & I \end{pmatrix}\begin{pmatrix} L_{11} & 0 \\ L_{21} & I \end{pmatrix} = \begin{pmatrix} I & 0_{12} \\ 0_{21} & I \end{pmatrix}$$

and equating the expressions for block matrix 0_{21}. For row i ($> k$), this decomposition yields $P^i_{21}L_{11} + L^i_{21} = 0$, i.e.,

$$\begin{aligned} p_{i1}l_{11} + p_{i2}l_{21} + \cdots + p_{ik}l_{k1} &= -l_{i1}\ , \\ p_{i2}l_{22} + \cdots + p_{ik}l_{k2} &= -l_{i2}\ , \\ \cdots\cdots\cdots\cdots\cdots\cdots & \\ p_{ik}l_{kk} &= -l_{ik}\ . \end{aligned}$$

The coefficients of this system are already available from preceding reduction steps. However, they bear accumulated round-off errors. Solution of this system would therefore only yield values p'_{ij} which approximate the true p_{ij}. Fortunately, that is all we need.

The preceding system, which is only of k-th order and triangular, is therefore solved recursively for the approximate solution $p'_{i1} \cdots p'_{ik}$. In a computer, the program for the "back solution" of the elimination procedure, already available within the machine, is effectively drawn upon for this purpose.

As shown above, the exact p_{ij}, applied to rows A^i, would yield the row S^i of very small elements, caused by cancellations in the most significant digits of each element. Instead, we take a linear combination with values p'_{ij} and replace the original matrix row A^i by

$$A'^i = p'_{i1}A^1 + \cdots + p'_{ik}A^k + A^i.$$

Similar cancellations of significant digits will again take place, leading to a row A'^i with small elements. This time, however, we shall take care to calculate with *double-precision*, furnishing small elements of A'^i that are *exact*.

Now we multiply through by a large number M to convert all elements to $O(1)$. Since the small elements were *exact*, no error is introduced or magnified by this step. The usual slump in accuracy has thus been completely prevented, maintaining the latter at its normal level.

The first k elements of the resulting row are now removed, and the row is thereby reduced to the form of R^i, by application of the elimination algorithm, utilizing the available rows $R^1 \cdots R^k$. It is in this reduction that the bulk of accumulation of round-off errors occurs. *Now*, however, the final R^i will have elements of $O(1)$, so that the effect of rounding errors will be no more appreciable than for all other normal rows.

Throughout k stages of reduction, row i ($> k$) has no effect on values in all other rows of R. Hence, apart from row i, the entire matrix R is retained as is.

We may look upon the reconditioning of matrix A in yet another light: When two matrices have their elements scaled such that corresponding row vectors have equal length (say all length 1), a comparison of their determinants provides an effective

comparison of "condition". Consider this done for the original matrix A (containing A^i) and for the modified A' (containing A'^i times the large number M). After complete reduction, we have $| A | = | L | \cdot | R | = \prod_{i=1}^{n} l_{ii}$. In the original system, a very small l_{ii} gave rise to a small $| A |$, indicating ill-condition. In the modified system, the element l'_{ii} in the identical position has been brought up to $O(1)$, carrying $| A' |$ to a comparatively high magnitude, which indicates favorable condition.

Summary. We summarize the auxiliary computations on encountering ill-conditioning: (1) recursive solution of a triangular linear system of order k; (2) double-precision recalculation of row A'^i, followed by a "left shift"; (3) elimination of the first k elements of row i.

It is noteworthy that removal of ill-conditioning requires the double-precision recalculation of only a single row and that intermediary results for all other rows are retained unaltered, thereby salvaging the bulk of computational effort.

Numerical example. We now illustrate the power of the present method by a numerical example. To simplify illustration, we shall desist from "positioning for size" for optimum divisors, but shall treat rows and columns in their original order.

Given the system

$$\begin{aligned}
1.32x_1 - 4.73x_2 + 5.39x_3 - 2.84x_4 + 3.97x_5 &= 1.33,\\
5.68x_1 - 6.25x_2 + 1.40x_3 + 7.02x_4 + 4.50x_5 &= -6.04,\\
1.93x_1 + 1.34x_2 - 2.16x_3 + 3.81x_4 - 2.62x_5 &= -1.75,\\
2.85x_1 + 3.09x_2 + 4.41x_3 + 2.36x_4 + 3.14x_5 &= 2.34,\\
4.32x_1 + 0.20x_2 + 4.69x_3 - 1.63x_4 - 5.39x_5 &= 4.50.
\end{aligned}$$

Suppose we stipulate that whenever, in any row, the element of maximum magnitude becomes less than 0.20, the entire row has fallen below an acceptable threshold of accuracy.

Applying straightforward elimination and rounding all figures to two decimal places, we obtain, after $k = 3$ stages, the system $R^{(3)}x = b^{(3)}$, i.e. $A = L^{(3)}R^{(3)}$, with the numerical values

$$(R^{(3)}, b^{(3)}) = \begin{bmatrix}
1 & -3.58 & 4.08 & -2.15 & 3.01 & 1.01\\
 & 1 & -1.55 & 1.37 & -0.89 & -0.84\\
 & & 1 & -1.21 & -0.39 & 1.17\\
 & & & 6.47 & 11.61 & -5.03\\
 & & & -0.08 & -0.01 & 0.02
\end{bmatrix},$$

$$L^{(3)} = \begin{bmatrix}
1.32 & & & & \\
5.68 & 14.08 & & & \\
1.93 & 8.25 & 2.76 & & \\
2.85 & 13.29 & 13.38 & 1 & \\
4.32 & 15.67 & 11.35 & 0 & 1
\end{bmatrix}.$$

The smallness of elements in row $i = 5$ now reveals the hitherto hidden fact that row 5 of A is almost a linear combination of the first $k = 3$ matrix rows. Writing L^T = transpose of the upper left-hand 3×3 submatrix L_{11} of $L^{(3)}$, we therefore solve the triangular system $L^T p^i + l^i = 0$, i.e.

$$1.32p_{51} + 5.68p_{52} + 1.93p_{53} = -4.32$$
$$14.08p_{52} + 8.25p_{53} = -15.67$$
$$2.76p_{53} = -11.35$$

still always rounding to two decimal places, yielding the approximate solution $p'_{51} = -2.86$, $p'^{1}_{52} = 1.30$, $p'_{53} = -4.11$. These values are utilized in replacing row 5 of the original augmented matrix (A, b) by the row

$$A'^5 = p'_{51}A^1 + p'_{52}A^2 + p'_{53}A^3 + A^5$$

calculated to double-precision, i.e. the fifth equation is replaced by

$$-0.0035x_1 + 0.0954x_2 - 0.0278x_3 - 0.0407x_4 - 0.1260x_5 = 0.0367$$

We now multiply through by a comparatively large number, say $M = 100$, and reduce the resultant first three terms to zero by subtracting multiples of rows 1, 2 and 3 of the matrix $(R^{(3)}, b^{(3)})$. At the end of stage 3, row 5 of the augmented matrix $(R^{(3)}, b^{(3)})$ now becomes

$$(0, 0, 0, -2.26, 0.31; -2.47),$$

in which the numerical value of the element of largest magnitude is now far above the permissible minimum threshold of 0.20.

Completion of numerical solution yields final values for the unknowns which are compared in the table below. Deviations δ_i from the true values x_i as well as $\max |\delta_i|$ and $\sum |\delta_i|$ are likewise tabulated. The improvement in accuracy is indeed striking.

Comparison of numerical values

Quantity	Correct values	Usual method	Present method	Quantity	Usual method	Present method
x_1	−2	−0.30	−1.96	δ_1	−1.70	−0.04
x_2	0	0.39	0.00	δ_2	−0.39	0.00
x_3	2	0.77	1.95	δ_3	1.23	0.05
x_4	1	−0.23	0.96	δ_4	1.23	0.04
x_5	−1	−0.31	−0.97	δ_5	−0.69	−0.03
$\max \lvert\delta_i\rvert$	0	1.70	0.05			
$\sum \lvert\delta_i\rvert$	0	5.24	0.16			
p	0	87	3			

Relative percentage deviations $100 \mid \delta_i \mid / \mid x_i \mid$ for individual components become infinite or unduly large whenever $\mid x_i \mid = 0$ or very small. In order to summarize the

overall relative inaccuracy by a single number, avoiding at the same time any instability in the neighborhood of small $|x_i|$, we define as an effective measure of overall inaccuracy

$$p = \text{overall percentage error} = 100 \sum |\delta_i| / \sum |x_i|$$

Numerical values of overall measure p, shown in the above table, likewise compare very favorably.

Closeness of solution. Substituting the numerical solution into the given set of equations, we find that the residual vectors for the two sets of unknowns are both very small and have elements of comparable magnitude. How then can we say that our second set of x_i is any "better" than the first?

Let x = exact solution vector, x' = computed solution vector, $\delta = x - x'$ = error vector for the unknowns, $r = b - Ax'$ = residual vector for the computed solution. Substitution for x' in the expression for r yields $r = A\delta$.

In each case we obtain a small vector r, but this cannot serve as a valid criterion for closeness of solution. Our prime objective is the reduction not of r but of $\delta = x - x' = A^{-1}r$, and this objective has been successfully achieved.