Error Analysis Revisited

J. H. WILKINSON, FRS, FIMA*
Formerly NPL, Teddington, Middlesex

0. Introduction

I HOPE it will not be regarded as too controversial if I start this paper with the categorical assertion that detailed rounding error analysis does not provide ideal bedside reading. Forty years' experience with the topic has convinced me that it is inherently tedious and boring and in any case is largely unnecessary.

To make matters worse the subject got off to an appallingly bad start. In the 1940's we convinced ourselves that Gaussian elimination was catastrophically unstable in spite of the fact that in the very form in which we were then using it (on desk machines) it is quite extraordinarily stable. In 1947 von Neumann and Goldstine[1] managed to allay our worst fears. However although their paper deservedly received widespread recognition very few have ever been able to give a concise sketch of the basis of their analysis. Why should a mastery of that paper have proved so elusive?

It is my view that the explanation is as follows. At the time when von Neumann and Goldstine wrote their paper there had been no previous systematic analysis of rounding errors or of the use of norms in matrix algebra. It was inevitable that these topics should have been discussed rigorously in that paper. Unfortunately they dominate it while no clear exposition of the basic strategy is included. It would be unreasonable to criticise this feature of their paper but there can be no doubt that it accounts for its "indigestibility." The reader is faced with page after page of detailed rounding error analysis and quite soon his senses are numbed. He may well emerge nursing the illusion that he has understood it in that he has followed the derivation of each line from the previous one, but there is more to "understanding" than that. Unfortunately we have all tended to follow the pattern established in that paper. Here I intend to "go the whole hog" and to avoid discussion of specific rounding procedures as much as possible.

1. Forward and backward error analysis

I shall refer to two basic forms of error analysis which I now define with reference to a strictly algebraic algorithm. Such an algorithm starts with a set of data elements $d_1, d_2, \ldots, d_r$ from which it determines via the fundamental arithmetic operations a sequence of intermediate values $e_1, e_2, \ldots, e_s$ leading finally to a set of solution elements $f_1, f_2, \ldots, f_t$. The relative magnitudes of r, s, t vary enormously.

(i) Classical forward analysis

In this one attempts to find bounds for $|\bar{e}_i - e_i|$ and hence finally for $|\bar{f}_j - f_j|$; usually the bounds will involve the computer precision. It is natural to define $\bar{f}_j - f_j$ to be "the error in f_j."

(ii) Backward error analysis

In this one attempts to determine a modified set of data elements $\bar{d}_i$ such that the computed set of $\bar{f}_j$ is the exact solution. There may be an infinite number of such sets; sometimes there is just one such set. It can happen even in connection with very simple algorithms that no such set exists. In any case one attempts to identify a specific data set related in a natural way to the steps involved in the algorithm.

There is no "conflict" between the two forms of analysis. I have used both regularly for many years and have possibly used the forward form more frequently than the backward form. The term "classical" forward analysis distinguishes it from other quite different forms of forward analysis which also happen to be very useful in linear algebra. (There is no time to discuss them here.) In the mid-1940's we all used classical forward analysis. We merely called it "rounding error analysis" and scarcely contemplated the possibility of using anything else.

The attitude of many to backward analysis is still rather bizarre. Some adopt a highly moral tone when discussing it, as though the user were guilty of moral turpitude. Others imagine that advocates of backward analysis are indifferent to the "accuracy" of their solutions or are unaware that the exact solution of a neighbouring problem may not be close to the solution of the given problem. This last criticism is the hardest to bear since enthusiasts for backward analysis are almost invariably keenly interested in perturbation theory and are therefore particularly unlikely to be guilty of that sin.

One of the greatest virtues of backward analysis (particularly in linear algebra) is that when it *is* the appropriate form of analysis it tends to be very markedly superior to forward analysis. Invariably in such cases it has remarkable formal simplicity and gives deep insight into the stability (or lack of it) of the algorithm.

2. Matrix equations

In this paper I shall concentrate on a set of algorithms that are of fundamental importance in numerical linear algebra. They are concerned with the solution of simple matrix equations. These algorithms provide particularly good material for the illustration of the effectiveness of backward error analysis.

In the first column of Table I sixteen such equations are listed. In each of them one entity (or in (xv) and (xvi), *two* entities) is to be regarded as the unknown, the other entities being regarded as given. In some cases the entity to be determined is a vector and in others it is a matrix. This entity is underlined and in most cases is denoted by $\underline{x}$ or $\underline{X}$. In (xv) and (xvi) two entities are to be determined from one equation and hence both are underlined. L is used throughout to denote a lower triangular matrix which may possibly have a unit

* Died October 5th, 1986.

Table I

Equation to be solved	Relation satisfied by computed entity
(*i*) $\underline{x} = c + Ab$	$\bar{x} - (c + Ab) = \Delta$
(*ii*) $\underline{X} = C + AB$	$\bar{X} - (C + AB) = \Delta$
(*iii*) $L\underline{x} = b$	$L\bar{x} - b = \Delta$
(*iv*) $\underline{x}^{\mathrm{T}}L = b^{\mathrm{T}}$	$\bar{x}^{\mathrm{T}}L - b^{\mathrm{T}} = \Delta$
(*v*) $U\underline{x} = b$	$U\bar{x} - b = \Delta$
(*vi*) $\underline{x}^{\mathrm{T}}U = b^{\mathrm{T}}$	$\bar{x}^{\mathrm{T}}U - b^{\mathrm{T}} = \Delta$
(*vii*) $L\underline{X} = I$	$L\bar{X} - I = \Delta$
(*viii*) $\underline{X}L = I$	$\bar{X}L - I = \Delta$
(*ix*) $U\underline{X} = I$	$U\bar{X} - I = \Delta$
(*x*) $\underline{X}U = I$	$\bar{X}U - I = \Delta$
(*xi*) $L\underline{X} = A$	$L\bar{X} - A = \Delta$
(*xii*) $\underline{X}L = A$	$\bar{X}L - A = \Delta$
(*xiii*) $U\underline{X} = A$	$U\bar{X} - A = \Delta$
(*xiv*) $\underline{X}U = A$	$\bar{X}U - A = \Delta$
(*xv*) $\underline{L}\,\underline{U} = A$	$\bar{L}\bar{U} - A = \Delta$
(*xvi*) $A\underline{L} = \underline{L}H$	$A\bar{L} - \bar{L}\bar{H} = \Delta$

diagonal. Similarly U denotes an upper triangular matrix which may possibly have a unit diagonal.

The equations vary enormously in their level of sophistication. The first two are concerned merely with matrix multiplication. The next four involve the solution of triangular sets of linear equations and the next four the computation of inverses of triangular matrices. Equation (*vii*) gives a right-hand inverse of L while equation (*viii*) gives a left-hand inverse. Of course with exact computation the two solutions would be identical. Similar remarks apply to (*ix*) and (*x*). Equation (*xv*) is the matrix equivalent of Gaussian elimination; both L and U are to be determined from the one equation. Similarly in (*xvi*) both L and H are to be determined; L is unit lower triangular and H is upper Hessenberg. The subdiagonal elements of L in the *first* column *could* be chosen arbitrarily, but usually they are taken to be zero. Since $H = L^{-1}AL$ this equation is the matrix equivalent of the reduction of A to upper Hessenberg form H by an elementary similarity transformation.

It is a remarkable fact that in spite of their disparate nature all of these matrix equations are solved in the same way. In every case we derive an equivalent set of scalar equations by equating in turn elements on the two sides. Thus from (*ii*) we have

$$x_{ij} = c_{ij} + \sum a_{ik}b_{kj} \tag{2.1}$$

for all relevant i, j. Notice that the matrices need not be square. All that is necessary is that they should be of dimensions which render the equation meaningful.

From equation (*iii*) we have

$$l_{i1}x_1 + \ldots + l_{ii}x_i = b_i \quad (i = 1, \ldots, n), \tag{2.2}$$

giving

$$x_i = (b_i - l_{i1}x_1 - \ldots - l_{i,i-1}x_{i-1})/l_{ii}. \tag{2.3}$$

With equations (*i*) and (*ii*) the unknown elements may be determined in any order since every x_{ij} is independent of all the others. Equation (*iii*) is different in that $x_1, x_2, \ldots, x_{i-1}$ *must* be determined before x_i.

Turning to equation (*xv*) the scalar equation takes a different form according as the (i, j) is below the diagonal or not. We have

$$l_{i1}u_{1j} + \ldots + l_{i,i-1}u_{i-1,j} + u_{ij} = a_{ij} \quad (i \leqslant j), \tag{2.4}$$

$$l_{i1}u_{1j} + \ldots + l_{i,j-1}u_{j-1,j} + l_{ij}u_{jj} = a_{ij} \quad (i > j). \tag{2.5}$$

In (2.4) we have taken account of the fact that $l_{ii} = 1$. Notice that there are precisely n^2 unknown elements in L and U combined. Equation (2.4) is used to determine u_{ij} via the relation

$$u_{ij} = a_{ij} - l_{i1}u_{1j} - \ldots - l_{i,i-1}u_{i-1,j}, \tag{2.6}$$

and equation (2.5) is used to determine l_{ij} via the relation

$$l_{ij} = (a_{ij} - l_{i1}u_{1j} - \ldots - l_{i,j-1}u_{j-1,j})/u_{jj}. \tag{2.7}$$

The order in which the elements of the entities L and U are determined is no longer unique but the elements of L and U on the right of equations (2.6) and (2.7) must be computed before the "new" element u_{ij} or l_{ij} is determined.

It is convenient at this point to describe the relation between the elements in the matrix factorisation and those derived by Gaussian elimination. In fact as the right-hand side of (2.6) is computed by subtracting the successive products we have (with an obvious notation)

$$\begin{aligned} &a_{ij}^{(1)} = a_{ij}, \quad a_{ij}^{(2)} = a_{ij} - l_{i1}u_{1j}, \\ &a_{ij}^{(3)} = a_{ij} - l_{i1}u_{1j} - l_{i2}u_{2j}, \ldots, \\ &u_{ij} = a_{ij}^{(i)} = a_{ij} - l_{i1}u_{1j} - l_{i2}u_{2j} - \ldots - l_{i,i-1}u_{i-1,j}. \end{aligned} \tag{2.8}$$

Similarly as the right-hand side of (2.7) is computed we have successively

$$\begin{aligned} &a_{ij}^{(1)} = a_{ij}, \quad a_{ij}^{(2)} = a_{ij} - l_{i1}u_{1j}, \\ &a_{ij}^{(3)} = a_{ij} - l_{i1}u_{1j} - l_{i2}u_{2j}, \ldots, \\ &a_{ij}^{(j)} = a_{ij} - l_{i1}u_{1j} - l_{i2}u_{2j} - \ldots - l_{i,j-1}u_{j-1,j}. \end{aligned} \tag{2.9}$$

Thus all of the elements which arise as the successive $a_{ij}^{(k)}$ in Gaussian elimination are derived as intermediate values as the right-hand sides of (2.6) and (2.7) are evaluated. The l_{ij} are, of course, the m_{ij} of Gaussian elimination. In the matrix formulation of Gaussian elimination we effectively derive all the values which occur in any (i, j) position at one time and then move on to the next (i, j) position.

In much the same way with equation (*xvi*) the (i, j) equation takes a different form according as $j \geqslant i - 1$ or $j < i - 1$, the former leading to the determination of the i, j element of H and the latter to the determination of the i, j element of L.

We observe now that for all sixteen of the matrix equations each element of the unknown entity or entities is derived via an equation of the form

$$q = (e \pm \sum f_k g_k)/d. \tag{2.10}$$

In equation (2.10) the q, e, f_k, g_k and d are generic symbols. Thus in equation (2.3)

$$q = x_i, \quad e = b_i, \quad f_k = l_{ik}, \quad g_k = x_k, \quad d = l_{ii}. \tag{2.11}$$

In some cases e may be zero; thus in solving (*vii*) most elements of the right-hand side I are zero and hence the e corresponding to these elements is absent in (2.10). In other cases $d = 1$ as, for example, in (2.3) when L is a unit lower triangle. The division operation is then omitted since there is no need to divide by 1.

3. Running error analysis

When the unknown entity is computed using inexact arithmetic it does not satisfy the matrix equation exactly.

I shall distinguish the computed entities by the use of bars. Thus the computed x obtained from equation (*iii*) is denoted by $\bar{x}$ and the computed L and U obtained from equation (xv) are denoted by $\bar{L}$ and $\bar{U}$. I have used two entirely different forms of backward error analysis to assess the computed entities.

The first I refer to as a *running* error analysis. It is particularly interesting in that it can be discussed in the first instance without *any* reference to the nature of the arithmetic being used. With this form of analysis we define in every case a residual entity Δ, as given in column 2 of Table I. Notice that when the unknown entity is a vector (matrix), Δ is a vector (matrix). If no errors were made Δ would be null in every case.

How are the elements of Δ related to the arithmetic errors made in computing the elements of the unknown entity? The answer is incredibly simple. The element Δ_j (or Δ_{ij}) is exactly equal to the sum of the arithmetic errors associated with equating the jth or (i, j)th element in the matrix equation. Thus in connection with (xv) equation (2.6) shows that there are $i-1$ subtractions and $i-1$ multiplications associated with the determination of u_{ij} when $i \leqslant j$ while equation (2.7) shows that there are $j-1$ subtractions, $j-1$ multiplications and one division associated with the determination of l_{ij} when $i > j$. Hence Δ_{ij} is exactly the sum of the errors made in the $2(i-1)$ arithmetic operations when $i \leqslant j$ while it is exactly the sum of the errors made in the $2(j-1)+1$ arithmetic operations when $i < j$.

This is not a first order result; it does not depend on the errors being small. It is true for example if all the elements involved are integers and the only errors are arithmetic blunders and it does not matter how large those errors may be.

In order to draw attention to the extraordinary simplicity and generality of the result we avoided defining what we meant by "the error made in an arithmetic operation." This is not quite as trivial as it might seem. Suppose we are adding 15 to 14 and we accidently take the answer to be 39 instead of 29 then the error is defined to be +10. Similar definitions apply to subtraction and multiplication. Thus if

$$\begin{aligned} &x = a + b, \\ &\bar{x} = \text{computed}(a+b), \text{ then error} = \bar{x} - (a+b) = \bar{x} - x, \end{aligned} \tag{3.1}$$

$$\begin{aligned} &x = a - b, \\ &\bar{x} = \text{computed}(a-b), \text{ then error} = \bar{x} - (a-b) = \bar{x} - x, \end{aligned} \tag{3.2}$$

$$\begin{aligned} &x = a \times b, \\ &\bar{x} = \text{computed}(a \times b), \text{ then error} = \bar{x} - (a \times b) = \bar{x} - x. \end{aligned} \tag{3.3}$$

For division we define the error rather differently. If

$$x = a/b,\ \bar{x} = \text{computed}(a/b) \text{ then error} = b\bar{x} - a. \tag{3.4}$$

It might have seemed more natural to define the error by

$$\text{error} = \bar{x} - x. \tag{3.5}$$

However the definition we use turns out to be more convenient and from the standpoint of backward analysis it is consistent with the other three. We are attempting to solve the simple matrix equation $bx = a$ and I have defined the error to be the residual.

I now turn to the point which appears to worry many when they first encounter backward error analysis of a matrix algorithm. When, for example, we attempt to compute x_i via equation (2.3) we have

$$\bar{x}_i = \text{computed}[(b_i - l_{i1}\bar{x}_1 - \ldots - l_{i,i-1}\bar{x}_{i-1})/l_{ii}]. \tag{3.6}$$

In other words $\bar{x}_i$ is computed using the previously computed $\bar{x}_j$ and these, too, are in error. The contribution to Δ_i resulting from multiplying l_{i1} by $\bar{x}_1$ must be determined from the error in that multiplication itself without reference to any error in $\bar{x}_1$. To be precise if $l_{i1} = 3$ and $\bar{x}_1$ is 4 and the product is taken to be 13 the error is +1. It may be that with exact computation x_1 should have been 5 and the product should have been 15. This is irrelevant and this is perhaps just as well since in practice we have $\bar{x}_1$ but we do not know what x_1 should have been. Backward error analysis establishes a relation between elements of the computed entity and the elements of the data entities. No reference of any kind is made to the exact solution of the relevant matrix equation.

It is perfectly natural to feel uneasy about this. Surely in equation (*iii*), for example, it is a bound for $\bar{x} - x$ which is ultimately required. In many cases this will be so but such a bound is derived subsequently by a *mathematical* error analysis. Thus if we know that

$$L\bar{x} = b + \Delta \quad \text{while} \quad Lx = b, \tag{3.7}$$

the Δ having been determined via backward error analysis, we can now set about determining a bound for $\bar{x} - x$. At this stage the fact that Δ has been obtained by a backward error analysis is irrelevant.

We now justify our assertion about the nature of Δ. We need only consider the prototype (2.10). We have (with the choice of the plus sign)

$$\bar{q} = \text{computed}[(e + \Sigma f_k g_k)/d], \tag{3.8}$$

To introduce our notation we write (for exact computation)

$$\begin{aligned} &s_0 = e, \quad p_k = f_k g_k \quad (k = 1, \ldots, r), \\ &s_k = s_{k-1} + p_k \quad (k = 1, \ldots, r), \end{aligned} \tag{3.9}$$

and for the corresponding computed quantities

$$\bar{s}_0 = e \quad \text{(no error made!)}, \tag{3.10}$$

$$\bar{p}_k = \text{computed}(f_k g_k) = f_k g_k + \mu_k \qquad (k = 1, \ldots, r), \tag{3.11}$$

$$\bar{s}_k = \text{computed}(\bar{s}_{k-1} + \bar{p}_k) = \bar{s}_{k-1} + \bar{p}_k + \alpha_k \qquad (k = 1, \ldots, r), \tag{3.12}$$

where (by definition) μ_k is the error made in the kth multiplication and α_k is the error made in the kth addition. [*N.B.*, α_k is *defined* to be the error made in attempting to add $\bar{p}_k$ to $\bar{s}_{k-1}$; the fact that these two quantities may be seriously in error is irrelevant.] Finally we have

$$\bar{q} = \text{computed}[\bar{s}_r/d], \tag{3.13}$$

giving

$$\bar{q}d = \bar{s}_r + \delta, \tag{3.14}$$

where by our definition δ is "the error in the division." Adding equations (3.10) to (3.14) and cancelling common members,

$$\bar{q}d = e + \sum f_k g_k + (\sum \mu_k + \sum \alpha_k + \delta) \quad (3.15)$$

and we have the desired result. *There is no interaction between the errors and the result is exact almost, as it were, by definition.* With the minus sign we have

$$qd = e - \sum f_k g_k + (-\sum \mu_k + \sum \sigma_k + \delta). \quad (3.16)$$

Here σ_k is the error in the kth subtraction. Notice that the contribution from the kth multiplication is $-\mu_k$; for simplicity we used the term "*sum* of the arithmetic errors" to cover both positive and negative contributions. When we turn to a detailed rounding error analysis we shall merely have an upper bound for each error and the signs are immaterial.

This one analysis applies to all sixteen matrix equations and any form of arithmetic. When applied to (3.6), for example, it gives

$$l_{ii}\bar{x}_i = b_i - l_{i1}\bar{x}_1 - \ldots - l_{i,i-1}\bar{x}_{i-1} + (-\sum \mu_k + \sum \sigma_k + \delta), \quad (3.17)$$

where the μ_k, σ_k and δ are the errors in the multiplications, subtractions and the divisions. This gives

$$b_i - l_{i1}\bar{x}_1 - \ldots - l_{ii}\bar{x}_i = \sum \mu_k - \sum \sigma_k - \delta = \Delta_i. \quad (3.18)$$

We stress once again that the fact that the $\bar{x}_1, \ldots, \bar{x}_{i-1}$ which we use are "in error" is irrelevant. Our object is to obtain a relation between the *given* L and the computed $\bar{x}$; the true solution of $Lx = b$ does not enter into the discussion! One further point should be stressed. The matrix L may have been determined by a previous computation in which case it will have a bar. We shall then be attempting to solve

$$\bar{L}x = b \quad (3.19)$$

and the Δ will, of course, be defined by

$$\bar{L}\bar{x} - b = \Delta. \quad (3.20)$$

Similarly b may have come from an earlier computation.

Thus if we are attempting to solve $Ax = b$ by triangular factorisation, we use in succession (xv), (iii) and (v) and the above analysis applies to the $\Delta^{(1)}$, $\Delta^{(2)}$, $\Delta^{(3)}$ defined by the equations

$$\bar{L}\bar{U} - A = \Delta^{(1)}, \quad (3.21)$$

$$\bar{L}\bar{y} - b = \Delta^{(2)}, \quad (3.22)$$

$$\bar{U}\bar{x} - \bar{y} = \Delta^{(3)}. \quad (3.23)$$

These are exact equations and using them in reverse order we derive

$$\bar{L}\bar{U}x - \bar{L}\bar{y} = \bar{L}\Delta^{(3)}, \quad (3.24)$$

$$\bar{L}\bar{U}\bar{x} - b - \Delta^{(2)} = \bar{L}\Delta^{(3)}, \quad (3.25)$$

$$(A + \Delta^{(1)})\bar{x} - b - \Delta^{(2)} = \bar{L}\Delta^{(3)}, \quad (3.26)$$

$$i.e., \quad A\bar{x} = b - \Delta^{(1)}\bar{x} + \Delta^{(2)} + \bar{L}\Delta^{(3)}. \quad (3.27)$$

It is surprising having regard to the utter triviality of these results that I should so frequently have been told that I have "overlooked" the errors in one or other of the computed entities. I have not! I think their very simplicity contributes in some strange way to the failure to understand them and it is my hope that divorcing the analysis from any consideration of rounding errors will make it clearer.

4. Rounding error analysis

Having proved our fundamental result its implications for computation using any specific form of arithmetic should be a trivial exercise. All we need is a bound for the error made in each of the basic arithmetic operations.

For floating point arithmetic with correct rounding or chopping the most convenient results for our present purpose are

$$\bar{x} = \mathrm{fl}(a \pm b) \quad \text{implies} \quad |\bar{x} - (a \pm b)| \leqslant \varepsilon |\bar{x}|, \quad (4.1)$$

$$\bar{x} = \mathrm{fl}(a \times b) \quad \text{implies} \quad |\bar{x} - ab| \leqslant \varepsilon |\bar{x}|, \quad (4.2)$$

$$\bar{q} = \mathrm{fl}(a/b) \quad \text{implies} \quad |\bar{q}b - a| \leqslant \varepsilon |a|, \quad (4.3)$$

with

$$\varepsilon \leqslant \beta^{1-t}/2 \quad \text{(rounding)}; \quad \varepsilon \leqslant \beta^{1-t} \quad \text{(chopping)}. \quad (4.4)$$

Notice that the error bound is given in terms of the computed result or, in the case of division, in terms of the dividend. Hence in every case it will be available in the computer at the time when the operation *is being performed*; it may not be available when the algorithm has been completed. For this reason I later called it a "running," or "*pari passu*," error analysis. For the basic prototype we have immediately

$$|\bar{q}d - e \pm \sum f_k g_k| \leqslant e[\sum |\bar{p}_k| + \sum |\bar{s}_k| + |\bar{s}_r|], \quad (4.5)$$

the last entry coming from the division. If e is absent or $d = 1$, then the corresponding entry on the right-hand side is absent.

In my opinion it is counterproductive now to write out what this basic result implies for each of the sixteen equations. It merely leads to a plethora of indigestible lower and upper suffices. For example, when doing running error analysis on the ACE at no time did I write down these expressions. I merely took an existing program (without any error analysis) and modified it as follows. As each arithmetic operation was performed I added the absolute value of the computed result (or of the dividend) into the accumulating error bound. In addition to simplicity this had the advantage when the matrices were sparse of giving a lower bound than is obtained via the explicit expressions. Where, taking advantage of sparsity, the program skipped an arithmetic operation, the corresponding contribution to the error bound was automatically omitted. By contrast the direct application of (4.1) to the addition or subtraction of zero still gives the contribution $\varepsilon |a|$ to the error bound!

If any reader finds that he cannot at this point obtain appropriate error bounds for any form of arithmetic for which he knows the bounds corresponding to the four basic arithmetic operations then he has not understood this paper at all. Either he or I(!) must start again.

An interesting case is when the computer can accumulate inner products in double the standard working precision. This was true for me on desk

machines, PILOT ACE, DEUCE and ACE. In this mode the error in a floating point multiplication was always zero and the error in an addition or subtraction was bounded by $2^{-2t}|\text{computed result}|$. The only error at single precision level was the final rounding of "$\bar{q}$." Again there was no need to repeat the basic analysis; in fact the programming was actually simpler since there were no contributions from the multiplications. In the case of the LU factorisation the result is quite staggering. Unless some elements of $\bar{U}$ are substantially larger than the elements of A the bounds for the elements of Δ defined by

$$\bar{L}\bar{U} - A = \Delta \tag{4.6}$$

are actually comparable with a single rounding error in a typical element of A. When many of the elements of $\bar{U}$ are smaller than the corresponding elements of A, as they often are for systematically ill-conditioned equations, then it may well happen that all the errors made in the course of solving the system affect the solution far less than the errors made in reading A into the computer!

5. The purpose of rounding error analysis

At this point I turn to a highly emotive topic. What is the purpose of a rounding error analysis? Views on this are highly subjective and for this reason I shall use the first person throughout this section.

There can be no doubt that *my* views are strongly influenced by the history of the subject in the years 1945–60 (roughly). Those of us who were interested at the time had initially thought that G.E. was highly unstable even if we had doubted whether the 4^{n-1} factor of Hotelling's analysis was realistic. Although the von Neumann and Goldstine paper had shed some light on it we were still a little unhappy about the cumulative effect of rounding errors. It was widely believed that for very large systems iterative methods were to be preferred because they work throughout with the original system and therefore avoid this dreaded cumulative effect. (This was nonsense; the case for iterative methods really rests on their ability to take full advantage of sparsity.) The method of deflation in connection with the eigenvalue/vector problem was also widely believed to be extremely unstable.

In the 1950's I solved many linear systems by G.E. and many eigenvalue problems using deflation. In every case I computed the residuals defined, respectively, by

$$b - A\bar{x} = r, \tag{5.1}$$

$$A\bar{x} - \bar{\lambda}\bar{x} = r. \tag{5.2}$$

For the second of these I was struck by the fact that r was always at noise level relative to A. (The $\bar{x}$ were, of course, normalised.) In fact the residuals were about as small as they could possibly have been for the precision of computation. *I was well aware that this did not imply a corresponding accuracy in the solution.*

After some years' experience of this I happened, almost by accident, to observe that (5.2) implies that

$$(A - r\bar{x}^{\mathrm{T}})\bar{x} = \bar{\lambda}\bar{x} \quad (\text{since } \bar{x}^{\mathrm{T}}\bar{x} = 1). \tag{5.3}$$

In other words $\bar{\lambda}$ and $\bar{x}$ were exact for a matrix $A - r\bar{x}^{\mathrm{T}}$ and since $\|r\bar{x}^{\mathrm{T}}\|_2 = \|r\|$, this meant that they were exact for a matrix differing from A at the noise level of the computer.

Encouraged by this I turned to equation (5.1) and observed that in an analogous way it could be written in the form

$$(A + r\bar{x}^{\mathrm{T}}/\bar{x}^{\mathrm{T}}\bar{x})\bar{x} = b \tag{5.4}$$

and that

$$\|r\bar{x}^{\mathrm{T}}/\bar{x}^{\mathrm{T}}\bar{x}\|_2 = \|r\|_2/\|\bar{x}\|_2. \tag{5.5}$$

I examined this ratio for all the solutions I could lay my hands on. Again it proved to be at noise level relative to A. It is interesting that because of the normalisation of the eigenvectors, the result was more obvious for (5.2) than for (5.1).

I must confess that I was staggered by this result. I took the view that it could scarcely happen by accident. There must be a reason why the computed solution was the exact solution of a problem so incredibly close to the original one. I therefore set about analysing G.E. with the explicit objective of showing that the final triangle corresponded exactly to a slightly perturbed A. It proved to be almost trivial, though, of course, I did the analysis in the language of G.E. and not of LU factorisation. Because I was motivated by my observation about the residuals I thought of it as a backward analysis! The outcome though was to show that

$$A - \bar{M}\bar{A}^{(n)} = \Delta, \tag{5.6}$$

where $\bar{M}$ is the unit lower triangular matrix formed from the multipliers and $\bar{A}^{(n)}$ is the final upper triangular matrix. The Δ is, of course, identical with that corresponding to an LU decomposition. Given my motivation it was natural to describe the result in the following terms.

"The $\bar{M}$ and $\bar{A}^{(n)}$ are exact for the matrix $A - \Delta$." (I realised, right from the start, that Δ_{ij} was exactly the sum of the arithmetic errors made in the i, j position.) Without that motivation I might have described and proved my result in entirely different terms. I might well have argued "If $\bar{M}$ and $\bar{A}^{(n)}$ were the exact results $A - \bar{M}\bar{A}^{(n)}$ would be null; can I obtain bounds for the elements of $A - \bar{M}\bar{A}^{(n)}$ in terms of computed values?" If I had approached it in this way I would have called it "residual analysis," though the matrix Δ is the same in both cases! For the sixteen matrix equations backward analysis and residual analysis are identical. However, it turns out that backward analysis is a more general concept; many highly successful backward analyses are in no sense residual analyses.

I have been at pains to stress the triviality of the analysis. The hurdle to be overcome was the disinclination one has to think in those terms, not the inherent difficulty. As a matter of fact I had done two backward error analyses in connection with polynomials some years earlier but those I had done by accident. It had not occurred to me that I was using a general purpose tool which could be of great value in numerical linear algebra.

Trivial though the analysis is the implications are enormous. It was immediately obvious that for most of the examples I had dealt with up to that time there was no point in looking for alternative algorithms in the hope that they would give better results. I had scarcely if ever been provided with data having more than six decimals. The error in the solution resulting from the "accumulation of the rounding errors" had always been much less

than that resulting from the inherent errors in the data. The analysis also showed clearly for the first time the relevance of pivoting. The underlying algebra of the analysis is the same whether pivoting is done or not; but the contributions to Δ_{ij} depend on the size of the $\bar{a}_{ij}^{(k)}$ (these are the $\bar{s}_k$ of my prototype) and the size of computed $(\bar{m}_{ik}\bar{a}_{kj}^{(k)})$, (these are the $\bar{p}_k$ of my prototype). If no pivoting is used these may become arbitrarily large even when A is "well conditioned." The purpose of pivoting is to minimise the possibility of growth.

The clear identification of the factors determining the stability of an algorithm soon led to the development of better algorithms. The proper understanding of inverse iteration for eigenvectors and the development of the QR algorithm by Francis are the crowning achievements of this line of research.

For me, then, the primary purpose of the rounding error analysis was insight. What then is the relevance of the analysis for the determination of "error bounds for the computed solution"? It should be appreciated first of all that this problem exists quite independently of the algorithm used to derive the solution. If we are given an alleged solution $\bar{x}$ derived, for example, by the evaluation (in inexact arithmetic) of a set of explicit formulae we may still wish to determine a bound for $\bar{x} - x$. Now

$$b - A\bar{x} = r \quad \text{implies that} \quad A(x - \bar{x}) = r \tag{5.7}$$

and hence that

$$x - \bar{x} = A^{-1}r. \tag{5.8}$$

We have therefore

$$\|\bar{x} - x\| = \|A^{-1}r\| \quad \text{(exactly)} \tag{5.9}$$

though we must remember that we have tacitly ignored the fact that r itself has to be "computed." We certainly have

$$\|\bar{x} - x\| \leq \|A^{-1}\|\,\|r\| \leq u\,\|r\| \tag{5.10}$$

if u is an upper bound for $\|A^{-1}\|$. This is a *rigorous* upper bound for the error but will it, in general, be a *realistic* upper bound? The answer is that if we consider all possible approximate solutions, good and bad, then except when A is very well conditioned, the probability is incredibly high that this will be a severe overestimate of the error, in fact it will usually be too large by a factor of the order of $\kappa(A)$, *i.e.*, $\|A\|\,\|A^{-1}\|$.

However, if $\bar{x}$ is determined by G.E. (or other backward stable direct methods) the errors in it are related in a very remarkable way. They are correlated so that $\|A^{-1}\|\,\|r\|$ is an *extremely* realistic error bound. This is so important that I would like to illustrate it by a trivial example. Suppose we have the 2×2 system

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{5.11}$$

with the $a_{ij} = 0(1)$ and $a_{11}a_{22} - a_{12}a_{21} = O(10^{-6})$. Here we have an ill-conditioned system with $\kappa(A) = O(10^6)$. If we solve this using 10 decimal floating point arithmetic, classical forward error analysis tells us (quite correctly) that the computed $\bar{x}_i$ will have only four correct figures unless, by a fluke, the rounding errors are very small. However, $\bar{x}$ will be very special. If we replace the last six digits in $\bar{x}_1$ and $\bar{x}_2$ by zeros (N.B., all these figures are wrong anyway, so we are not degrading the quality of the solution!), the residual will be increased by a factor $O(10^6)$. In fact if we replace the incorrect figures with random digits the probability is almost unity that the residuals will be $O(10^6)$ times as large as those corresponding to $\bar{x}$ itself. Why is this so? It is precisely because G.E. is backward stable, *i.e.*, x is the exact solution of

$$(A + \Delta)\bar{x} = b \tag{5.12}$$

where, in our case, $|\Delta| = O(10^{-10})$. This remarkable property of the residuals extends to almost all the backward stable algorithms in linear algebra. Thus, for example, when the basic vectors $[x_1 x_2 \ldots x_r] = X$ of an invariant subspace are found by a backward stable algorithm the errors in the computed $\bar{X}$ will be specially related in such a way as to make it possible to get realistic error bounds for $\|\bar{X} - X\|$ with an economy of effort that is just not possible for an $\bar{X}$ of comparable accuracy determined by an algorithm which is not backward stable.

It sometimes happens that one requires an "improved" solution to a linear system and a bound for the error in the corrected solution. This can be achieved if we have an approximate inverse $\bar{X}$. Thus if

$$A(x - \bar{x}) = b - A\bar{x} = r, \tag{5.13}$$

we have

$$\bar{X}A(x - \bar{x}) = \bar{X}r. \tag{5.14}$$

If $\bar{X}$ is an approximate L.H. inverse determined via LU decomposition the backward stability of this algorithm again ensures that $\bar{X}A - I$ is far smaller than would be true of a rival $\bar{X}$ of the same accuracy. (Indeed, smaller by a factor of order $\kappa(A)$.) Hence writing

$$\bar{X}A = I - E \tag{5.15}$$

we have

$$\begin{gathered} (I - E)(x - \bar{x}) = \bar{X}r \\ \|x - \bar{x} - \bar{X}r\| \leq \|E\|\,\|\bar{X}r\|/(1 - \|E\|). \end{gathered} \tag{5.16}$$

This gives an extremely realistic bound for the error in the corrected solution $\bar{x} + \bar{X}r$.

Although it was realised well before backward error analysis was established that error bounds for computed solutions could be derived by methods of the type described above it is not generally appreciated even now how large a part backward stability plays in making it possible to get *realistic* error bounds by very simple means.

There is one further aspect of backward stability that needs to be stressed. It is not at all uncommon for the primary objective in solving a system

$$Ax = b \tag{5.17}$$

to be the determination of an $\bar{x}$ such that $A\bar{x}$ reproduces b accurately and $x - \bar{x}$ may be more or less irrelevant. Now in general, although the exact x reproduces b exactly, there will be no machine representable $\bar{x}$ which does so. Because G.E. is backward stable $\bar{x}$ will reproduce b almost as accurately as the correctly rounded version of x. Similar considerations often apply in connection with the more difficult problems such as computing invariant subspaces. At intermediate stages in

a long computation the degree of inner consistency may be far more important than the absolute accuracy of the computed entities.

6. *A priori* error analysis

The first backward error analysis in linear algebra that I completed was a running analysis of the type discussed in previous sections, but shortly afterwards I did a backward analysis of a different type. Viewed in retrospect I find it strange that I did not at the time explicitly realise that I was using two quite different types of analysis.

I first used the alternative form, which I call *a priori* analysis, in connection with the solution of a triangular system

$$Lx = b. \tag{6.1}$$

Here it had the irresistible attraction of showing that in floating point arithmetic the computed solution $\bar{x}$ satisfies exactly a perturbed system

$$(L + \delta L)\bar{x} = b \tag{6.2}$$

where the elements of L satisfy a relation of the kind

$$|(\delta L)_{ij}| \leqslant f(i, j)\beta^{-t}|L_{ij}|. \tag{6.3}$$

Here $f(i, j)$ is a fairly innocuous function. Note that we have componentwise bounds for the perturbation matrix δL. In the running error analysis we obtained a result of the form

$$Lx = b + \delta b. \tag{6.4}$$

We have written δb instead of Δ in conformity with the notation in (6.2). Our running analysis did not give satisfactory componentwise bounds for δb relative to b and indeed such bounds do not exist. The bounds given by running analysis are sharper than those given by *a priori* analysis; although the latter is more satisfactory in other respects. Unfortunately *a priori* analysis cannot be presented in a form which is independent of the arithmetic being used. For convenience we give the analysis appropriate to floating point arithmetic with correct rounding or chopping.

For the basic arithmetic operations we have

$$\bar{x} = \text{fl}\,(a \pm b) = (a \pm b)(1 + \theta) = x(1 + \theta), \tag{6.5}$$

$$\bar{x} = \text{fl}\,(a \times b) = ab(1 + \theta) = x(1 + \theta), \tag{6.6}$$

$$\bar{x} = \text{fl}\,(a \div b) = (a/b)(1 + \theta) = x(1 + \theta), \tag{6.7}$$

where

$$1/(1 + \varepsilon) \leqslant 1 + \theta \leqslant 1 + \varepsilon, \tag{6.8}$$

$$\varepsilon \leqslant \beta^{1-t}/2 \text{ (rounding)}; \quad \varepsilon \leqslant \beta^{1-t} \text{ (chopping)}. \tag{6.9}$$

Notice that

$$1/(1 + \varepsilon) \leqslant 1/(1 + \theta) \leqslant 1 + \varepsilon, \tag{6.10}$$

i.e., $1/(1 + \theta)$ satisfies the same relation as $1 + \theta$. For conciseness we shall sometimes replace $1 + \theta$ by ϕ so that we have

$$1/(1 + \varepsilon) \leqslant \phi \leqslant (1 + \varepsilon); \quad 1/(1 + \varepsilon) \leqslant 1/\phi \leqslant (1 + \varepsilon). \tag{6.11}$$

In our analysis we shall often add suffices to θ and ϕ in an obvious way.

In order to illustrate the essential difference between running analysis and *a priori* analysis we consider the computation of

$$s_r = a_1 + a_2 + \ldots + a_r. \tag{6.12}$$

For the running analysis we obviously have

$$|\bar{s}_r - s_r| \leqslant \varepsilon[|\bar{s}_2| + |\bar{s}_3| + \ldots + |\bar{s}_r|], \tag{6.13}$$

where the $\bar{s}$ are the computed partial sums. (This is even simpler than our prototype.) For the *a priori* analysis we write

$$\bar{s}_1 = a_1 \text{ (no error)}; \quad \bar{s}_k = \text{fl}\,(\bar{s}_{k-1} + a_k) = (\bar{s}_{k-1} + a_k)\phi_k, \tag{6.14}$$

giving typically when $r = 5$

$$\bar{s}_5 = a_1\phi_2\phi_3\phi_4\phi_5 + a_2\phi_2\phi_3\phi_4\phi_5 + a_3\phi_3\phi_4\phi_5 + a_4\phi_4\phi_5 + a_5\phi_5, \tag{6.15}$$

where each ϕ_i satisfies a relation of the form

$$1/(1 + \varepsilon) \leqslant \phi_i \leqslant 1 + \varepsilon. \tag{6.16}$$

Clearly we have

$$|\bar{s}_5 - s_5| \leqslant |a_1|\,d_4 + |a_2|\,d_4 + |a_3|\,d_3 + |a_4|\,d_2 + |a_5|\,d_1, \tag{6.17}$$

where

$$d_k \leqslant (1 + \varepsilon)^k - 1 \leqslant k\varepsilon(1 + \varepsilon)^{k-1}. \tag{6.18}$$

The result in the general case is obvious. We see that the bound we have for $|\bar{s}_r - s_r|$ is given in terms of the $|a_i|$ and hence we obtain the same bound for the sum $\pm a_1 \pm a_2 \pm \ldots \pm a_r$ whatever the distribution of signs. Clearly our new analysis takes no advantage of partial cancellations that may take place during the addition. (Running error analysis is expressed in terms of the computed $\bar{s}_k$ and hence *does* take account of cancellations.) Notice that we may write

$$|\bar{s}_5 - s_5| \leqslant (1 + \varepsilon)^4\varepsilon[4|a_1| + 4|a_2| + 3|a_3| + 2|a_4| + |a_5|], \tag{6.19}$$

where we have slightly weakened the result. The factor associated with $|a_1|$ is anomalous. With a further weakening we have

$$|\bar{s}_5 - s_5| \leqslant (1 + \varepsilon)^4\varepsilon[5|a_1| + 4|a_2| + 3|a_2| + 2|a_2| + |a_1|], \tag{6.20}$$

and we now have uniformity in the coefficients. Finally with a further weakening we have

$$|\bar{s}_5 - s_5| \leqslant 5(1 + \varepsilon)^4\varepsilon[|a_1| + |a_2| + |a_3| + |a_4| + |a_5|]. \tag{6.21}$$

The weakening effect in this last result could be substantial of course. We have given the analysis for the case when $r = 5$ since the extension to the general case is obvious and the main obstacle to reading an analysis is, (in my opinion), an excess of unimportant detail.

If we return to (6.15) it states that the computed sum is the exact sum of perturbed elements $a_1\phi_2\phi_3\phi_4\phi_5, \ldots, a_5\phi_5$. The factors associated with the a_i represent (for any reasonable precision of computation) small relative errors. We see then that however much (or little) cancellation takes place the computed sum is the exact sum of perturbed terms each differing from the true terms by low relative errors. We have small

componentwise perturbations, a most satisfactory feature.

It might well be asked what purpose the suffices have served. Could we not have written

$$\bar{s}_5 = a_1\phi^4 + a_2\phi^4 + a_3\phi^3 + a_4\phi^2 + a_5\phi, \quad (6.22)$$

where ϕ^k represents the product of k factors (in general all different), each satisfying the inequalities (6.11)? We can indeed do this and in practice often do. However, we must remember that

$$\phi^p \times \phi^q = \phi^{p+q}; \quad \phi^p/\phi^q = \phi^{p+q}. \quad (6.23)$$

The first of these causes no anxiety. The second is at first sight disconcerting. However, remember that all we know of the numerator and denominator is that

$$(1+\varepsilon)^{-p} \leqslant \phi^p \leqslant (1+\varepsilon)^p, \quad (1+\varepsilon)^{-q} \leqslant \phi^q \leqslant (1+\varepsilon)^q, \quad (6.24)$$

and hence all we can say of the quotient is that

$$(1+\varepsilon)^{-(p+q)} \leqslant \phi^p/\phi^q \leqslant (1+\varepsilon)^{p+q}, \quad (6.25)$$

and hence we require ϕ^{p+q} to cover the possible range of values. One soon gets used to this feature but it has one weakness. Although

$$\phi/\phi = \phi^2 \quad (6.26)$$

it is clearly true that

$$\phi_i/\phi_i = 1, \quad (6.27)$$

since ϕ_i denotes a specific value in the range $(1+\varepsilon)^{-1}$ to $1+\varepsilon$. In some cases we wish to take advantage of this last fact. Consider for example the relation

$$x_4 = \text{fl}\,[(b_4 - l_{41}x_1 - l_{42}x_2 - l_{43}x_3)/l_{44}]. \quad (6.28)$$

If we use suffices we find that

$$l_{44}x_4 = b_4\phi_2\phi_4\phi_6\phi_7 - l_{41}x_1\phi_1\phi_2\phi_4\phi_6\phi_7 - l_{42}x_2\phi_3\phi_4\phi_6\phi_7 - l_{43}x_3\phi_5\phi_6\phi_7, \quad (6.29)$$

where the ϕ_i's have been introduced in the natural order as each of the arithmetic operations is performed. This is an exact equation though we do not know where the ϕ_i lie within the standard range. Now it is convenient to get rid of the factor associated with b_4 by dividing by $\phi_2\phi_4\phi_6\phi_7$. Since we have used suffices we can cancel in the obvious way and *then* drop suffices to obtain (after rearrangement)

$$b_4 = l_{41}x_1\phi + l_{42}x_2\phi^2 + l_{43}x_3\phi^3 + l_{44}x_4\phi^4. \quad (6.30)$$

[*N.B.*, in (6.30) any factor of the form $1/\phi_k$ is replaced by ϕ.] If we had not used suffices we would not have known where cancellations occurred and the powers of ϕ would have been unnecessarily large.

Applying the above result to the solution of

$$Lx = b \quad (6.31)$$

we see that the computed $\bar{x}$ satisfies

$$(L + \delta L)\bar{x} = b \quad (6.32)$$

where, using the notation of (6.18),

$$|\delta L_{ij}| \leqslant |L_{ij}|d_j. \quad (6.33)$$

Notice that there is no perturbation associated with the b_i. It was to achieve this desirable state of affairs that we divided the typical equation (6.29) by the factor originally associated with b_4. In matrix form (6.33) implies that

$$\begin{aligned} |\delta L| &\leqslant |L|\,\text{diag}\,(d_j) \\ &\leqslant \varepsilon|L|\,\text{diag}\,[j(1+\varepsilon)^{j-1}] \\ &\leqslant \varepsilon(1+\varepsilon)^{n-1}|L|\,\text{diag}\,(j), \end{aligned} \quad (6.34)$$

where in the final line we have weakened the result slightly by replacing all $(1+\varepsilon)^{j-1}$ by $(1+\varepsilon)^{n-1}$.

It might reasonably be objected that the use of the term *a priori* to describe this form of analysis is unsatisfactory as it is mostly used in composite analyses to give *a posteriori* results!

Again each of the sixteen matrix equations can be analysed in exactly the same way and the reader should not have any serious difficulty in deriving the appropriate results. It should be emphasised that we do not have componentwise bounds in connection with the more sophisticated of the matrix equations. Thus for (xv), *i.e.*, $LU = A$, we obtain a relation of the form

$$\bar{L}\bar{U} - A = \Delta, \quad (6.35)$$

where

$$|\Delta| \leqslant |\bar{L}|\,\text{diag}\,(d_j)\,|\bar{U}|. \quad (6.36)$$

This means that when $\bar{L}$ and $\bar{U}$ have been determined, we can *a posteriori* determine a bound for $\|\Delta\|$ in a very simple manner, though it is not as sharp as that given by a running analysis. Moreover, when $\bar{L}$ and $\bar{U}$ are computed using accumulation of inner products in double precision and an approximate left-hand inverse $\bar{X}$ of A is then determined using this $\bar{L}$ and $\bar{U}$ (again with accumulation of inner products), analysis of the type used in the section gives a very good bound for

$$\|\bar{X}A - I\| \quad (6.37)$$

with very little extra computation. On ACE accumulation of inner products was achieved without a time penalty and this was easily the most efficient way of getting a realistic error bound for a corrected solution $\bar{x} + \bar{X}r$ using the mathematical analysis embodied in (5.14) to (5.16).

7. Recommendations

There are no conceptual difficulties in designing procedures for determining rigorous error bounds for computed solutions to most of the standard problems of numerical linear algebra, though for the more difficult eigenvalue problems (invariant subspaces, etc.) they are rather tedious. I have tended to regard them as intellectual exercises, the main purpose of which is to "sharpen one's wits." For an aspiring error analyst they provide experience comparable to that derived at an earlier stage in one's career from proving complicated, but elementary, trigonometric identities.

I doubt whether the production of a suite of NAG programs (*i.e.*, with all the appropriate robustness) could be justified when there are so many more urgent things to do. Much more useful would be a suite of programs which are the analogues of "iterative refinement." Such programs are quite simple to design and are efficient enough to be widely usable. They produce results of

extraordinary dependability. Indeed my experience with constructing truly rigorous programs and iterative refinement programs has been that any disagreements in their pronouncements have always been resolved in favour of the latter. A good working policy has been to continue to look for bugs in the rigorous program as long as it contradicts results given by the corresponding iterative refinement program!

Reference

1. von Neumann, John, and Goldstine, H. H., *Bull. Amer. Math. Soc.*, 1947, **53,** 1021–1099.