BULLETIN (New Series) OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 19, Number 2, October 1988

A CRITIQUE OF NUMERICAL ANALYSIS

PETER LINZ

1. Introduction. In this essay I want to raise the question "Is numerical analysis useful?". Most mathematicians, even those without any involvement in numerical computations, will think that the answer is obviously yes. It is common knowledge that computational methods are used daily by many members of the scientific community to solve problems that otherwise defy treatment. How can one seriously question the usefulness of something that has become a standard tool for many people?

To give substance to the question let me make a semantic distinction. I use the term *computational mathematics* to denote the wide spectrum of activities having to do with the approximate solution of scientific problems expressed through mathematical models. Typically, the equations arising from these models are differential or integral equations with no known closed form solution. For an approximate solution they must be discretized, that is, replaced by some finite system of equations that can be solved by algebraic methods. The whole process involves several phases and some quite distinct aspects. One is *numerical methodology* which considers ways of discretizing differential and integral operators and how best to solve the resulting finite systems. Another is *numerical analysis* which involves the rigorous study of the algorithms created by the methodology. The primary goal of analysis is to describe the relationship between the exact solution of the original equation and the approximate one obtained from its discretized version. It is numerical analysis in this narrower sense that I wish to examine here.

Even with this narrowed interpretation, the usefulness of numerical analysis is rarely questioned. Those who work in this area point out, with a great deal of justification, that analysis gives much insight into the nature of numerical methods and has contributed significantly to the widespread acceptance of numerical methodology. While some computational methods, such as relaxation and finite element techniques, were originated by engineers relying on physical insight, later analysis was crucial. Methods limited to special problems became general approaches as our theoretical understanding increased. In other instances the analysis suggested new methodologies. Numerical analysis has been instrumental in the design of effective numerical algorithms, and the effort expended has been repaid handsomely through the creation of a powerful tool for the solution of many important problems. Nevertheless, as I want to point out, this is not the end of the story. There are some fundamental issues that have been studied less thoroughly then they deserve, issues that grow in importance as scientists tackle more complex problems. There are open

Received by the editors February 23, 1988.
1980 *Mathematics Subject Classification* (1985 *Revision*). Primary 65-02.

problems here whose solution would have a major impact on computational mathematics. I will suggest some promising directions for further work, although I do not mean to imply that these are the only ones worth pursuing. What I want to emphasize is that there are unresolved questions in numerical analysis whose solutions call for a great deal of mathematical knowledge and ingenuity and which constitute a challenging and attractive research area.

The discussion will concern itself mostly with very fundamental and broad issues, which are "soft" in the sense that one cannot prove many theorems about them. But if we want to claim that numerical analysis is truly useful we must look at all relevant questions, even when immediate and very precise answers are not yet available.

2. Convergence analysis. First, let us look at a sort of capsule summary of the accepted modus operandi of numerical analysis. Modern numerical analysis tends to employ the terminology of functional analysis. In articles published in typical journals such as the SIAM Journal on Numerical Analysis or Numerische Mathematik, one routinely finds terms like Hilbert spaces, compact closure, and weak convergence. These concepts serve the theoretician well and have allowed the establishing of a coherent and extensive theory of the approximate solution of operator equations. While most articles on numerical analysis are technically difficult for those with only a modest knowledge of mathematics, one can extract some fairly simple general principles. I will illustrate this with the linear operator equation

$$Lx = y, \tag{1}$$

where $L\colon X \to Y$ is a linear operator between the normed linear spaces X and Y. The right side y is given and the equation is to be solved for the unknown x. We assume that L has a bounded inverse on Y. Many important scientific problems whose mathematical formulation involves linear differential and integral equations fall into this class. There are problems that do not fit into this abstract framework, but what I say here about (1) has its counterpart for most other numerical problems.

In the process of discretization, equation (1) is replaced with a parametrized sequence of problems

$$L_n x_n = y_n, \tag{2}$$

where now L_n is an operator on some n-dimensional spaces X_n and Y_n. For simplicity, let us assume that these spaces are subspaces of their counterparts X and Y. The quantity n is called the *discretization parameter*; it measures the degree to which the discretized operator L_n represents the original operator L.

Since L_n is effectively an $n \times n$ matrix, it is in principle possible to solve (2) algebraically. Ignoring such additional difficulties as round-off error in computer arithmetic, solving (2) then gives the approximate solution x_n. The fundamental concern of all analysis is the relation between the true solution x and its approximation x_n. In particular, we would like to prove *convergence*. By this we mean that, as we increase n, the approximate solution should come closer and closer to the true solution, in the sense that

$$\lim_{n\to\infty} \|x - x_n\| = 0.$$

Usually, the first step of the analysis is to define a *consistency* error

$$r_n(x) = Lx - L_n x. \tag{3}$$

From (1) and (2) we see immediately that

$$L_n(x - x_n) = y - y_n - r_n(x),$$

and we get a bound for the error of the approximate solution

$$\|x - x_n\| \le \|L_n^{-1}\| \left\{\|y - y_n\| + \|r_n\|\right\}. \tag{4}$$

In most cases it is relatively elementary to show that

$$\lim_{n\to\infty} \|y - y_n\| = 0, \tag{5}$$

and that, under well-defined conditions,

$$\lim_{n\to\infty} \|r_n\| = 0. \tag{6}$$

We need one further result, the *stability* condition

$$\lim_{n\to\infty} \|L_n^{-1}\| \le K < \infty. \tag{7}$$

If we put this into (4), we get the central theorem of numerical analysis that stability and consistency imply convergence.

This, in a nutshell, is what conventional numerical analysis is all about. Because of technical difficulties, there are lots of unresolved problems, but most of the difficulties in analysis come in verifying the stability condition (7). Generally, this is not easy at all. There are other questions, for example how difficult it is to solve (2) or how quickly x_n coverges to x, but normally these are simpler to deal with than stability.

Suppose now that we have shown that a numerical method is stable and convergent. What does this tell us? It does show that, in some asymptotic sense, the method works. If the assumptions needed to justify (5) and (6) hold, then (4) and (7) tell us that we can in principle achieve arbitrarily high accuracy by making n sufficiently large. Before we start writing a computer program for some method, it is reassuring to know this (even though programs for most complicated practical problems are written without the benefit of a proof of convergence). Actually, we can often say more. In most cases the approximation error can be bounded more explicitly as

$$\|x - x_n\| \le n^{-p} \|L_n^{-1}\| \eta(x), \tag{8}$$

where η is some complicated functional of the unknown solution x. The quantity p is called the *order of convergence* and tells us something about how accurate the method is likely to be. For a given amount of computational effort, a method with a high order of convergence tends to give better results than one with a low order. This is balanced by the fact that high order methods usually are more restricted in their applicability and more difficult to implement. While there are a number of less obvious factors involved in (8), this kind of reasoning is often used and is a reliable guideline for the selection of numerical methods. Thus, a major usefulness of convergence analysis is as a plausible predictor for the success of a suggested algorithm. This is a very valuable aspect.

But there are other important concerns that (8) does not address. In particular, it may not give us a useful bound for the error $\|x - x_n\|$. The reason for this is two-fold. First, η involves the unknown solution x, so to bound $\|x - x_n\|$ we may need to know a good deal about x; for example, bounds for some of its higher derivatives. In most practical situations, this is nearly impossible to get. Second, even if η can be bounded, the various inequalities and estimates involved make the bound (8) quite unrealistic, overestimating the actual error by several orders of magnitude. Consequently, (8) is rarely used in practice. While there are many computer programs for solving complicated equations efficiently, few of them provide a rigorous assessment of the error.

3. Useful error analysis. Suppose we now set ourselves a goal that goes beyond the traditional convergence analysis as exemplified by (8), to get error bounds that can actually be used in practical situations. If such error bounds are to be useful, they must be both *computable* and *realistic*. These adjectives, while quite intuitive, do not lend themselves to a very precise definition. They are of course just matters of degree. An error bound is computable if all terms needed for it can be obtained with a manageable amount of effort; it is realistic if it does not overestimate the actual error by too much. An error bound which is too high by 50 percent may be considered realistic, one which overestimates by a factor of 100 surely is not. Still, by any commonly accepted standards, the bound given by (8) is in most situations neither computable nor realistic. But if we look for better alternatives, we find little in the literature of numerical analysis that will help us. There does not exist at present any systematic and general way of finding computable and realistic error bounds. A potential exception is the method of *a posteriori* error analysis, but even this has not been worked out to any significant extent.

The bound provided by (8) is of limited use because it involves the unknown x. We can avoid the difficulty by making an a posteriori analysis which uses instead the computed solution x_n. Manipulating (1) and (2) in a slightly different way, we can see easily that

$$\|x - x_n\| \leq \|L^{-1}\| \, \|\rho_n\|, \tag{9}$$

where ρ_n is the *residual* of the computed solution

$$\rho_n(x_n) = Lx_n - y.$$

It is often not hard to produce a reasonably good bound for $\|\rho_n\|$; we simply plug the computed solution back into (1) to see how well it fits. Putting a realistic bound on $\|L^{-1}\|$ is more difficult and is the key element on which the success of the a posteriori analysis depends.

The a posteriori method is quite old and what I have said here is elementary and well known. What I find surprising is that it is used so little. I have been unable to find any reference to it in introductory numerical analysis texts, and even more advanced treatments, such as Isaacson and Keller [**2**], refer to it only casually. The reason for this seems to lie in the perception that getting good bounds on $\|L^{-1}\|$ is impossible. The problem is certainly not a trivial one, but there are some recent indications that progress is possible. Analysis done on finite element methods, for example [**1**, **9**], has had some success for elliptic partial differential equations. In my own work I have used a combination of analytical and numerical techniques for the solution of integral equations [**4**, **5**]. Since many partial differential equations have integral equation analogues, this has some promise for future progress. But certainly much more work needs to be done and the finding of effective methods for bounding $\|L^{-1}\|$ for the most common equations of mathematical physics remains a challenging open problem.

I hasten to add that I do not see the a posteriori method as a panacea for numerical computation. It certainly has the attractive feature of conceptual simplicity, important if it is to be used in practice. It holds enough promise to make its further pursuit worthwhile, but it is unlikely that it can be used everywhere. There are other approaches that remain to be investigated or even invented. We do not know exactly where this will lead, but what is important is to recognize the existence of a serious problem in computational mathematics. The lack of realistic and computable error bounds prevents the design of very high-quality numerical software and makes it difficult for the user of existing software to view the results with a very high degree of confidence. This lack of reliability causes much wasted time and effort and on occasions results in undetected erroneous answers. There is an urgent need in computational mathematics for effective methods of obtaining useful error bounds. Success in this direction would have immediate drastic effects on how scientists solve numerical problems. Without significant progress, numerical computing will remain in its present unreliable state.

I am optimistic that some progress can be make and that a concerted effort in this direction will yield rich results. Nevertheless, it would be unrealistic to expect that rigorous and useful error bounds can be obtained for all problems. There will always be those problems that we are just barely able to solve and for which we have no hope of any significant analysis. What do we do in these cases? Here we enter a third area of computational mathematics, that of *numerical pragmatics.* This deals with the problem of making practical decisions where precise theoretical guidance is not available.

4. Numerical pragmatics. Many people, some of them with rather weak backgrounds in mathematics, solve problems numerically. This is possible because, on the whole, numerical methodology is easy to use. The discretization of even very complex partial differential equations by finite difference or finite element methods is relatively routine. Often the most challenging problem is the management of details involved in representing regions, finite elements, sparse matrices, and so on. In some situations stability problems arise, but there the stability analysis for simple problems gives enough guidance for the successful treatment of more complicated ones. With a little tinkering, an experienced engineer can solve most of his problems numerically. By solving I mean that he can get some reasonable numbers out of the computer as long as he does not worry too much about how good these results are.

How do engineers deal with the problem of assigning some measure of reliability to the numbers that the computer produces? Over the years, I have sat on many Ph.D. qualifying examinations or dissertation defenses for engineering students whose work involved a significant amount of numerical computing. In one form or another, I invariably ask two questions "Why did you choose that particular algorithm?' and "How do you know that your answers are as accurate as you claim?". The first question is usually answered confidently, using such terms as "second-order convergence" or "von Neumann stability criterion". The next question, alas, tends to be embarrassing. After an initial blank or hostile stare, I usually get an answer like "I tested the method with some simple examples and it worked", "I repeated the computation with several values of n and the results agreed to three decimal places", or more lamely, "the answers looked like what I expected". So far, I have not faulted any student for the unsatisfactory nature of such a response.

One reason for my reluctance to criticize is that I have really nothing better to offer. Rigorous analysis is out of the question. Even if the average engineering student were made to master the intricacies of functional analysis, what he would find would be disappointing. When we leave the realm of text-book problems, the requirements of most convergence analyses are too restrictive to be applicable, and the results in any case are not very helpful. In face of this failure of analysis, the student can hardly be blamed for resorting to simple and often successful rules of thumb. What I do find disturbing is the pragmatics that are used are often ill-considered. Take for example the common practice of repeating the computations with several values of the discretization parameters. The reasoning behind this is that, if the method converges and we observe that the solution has "settled down" in the first few decimal digits, we can be confident that it is actually exact to this accuracy. Sometimes this makes good sense, but unfortunately it does not always work. For example, it may not catch systematic errors such as a wrong sign somewhere or a dropped factor of two. But such errors are very common, particularly in the computer programs that are eventually written. There are many instances of programs that delivered incorrect results for a considerable period of time before the error was found. Still, one cannot blame the students too much. The subject of numerical pragmatics is rarely mentioned in numerical analysis texts and is, in my experience, hardly ever treated satisfactorily in courses on the subject.

Pragmatics, being by nature imprecise, can always fail, but they should not be ignored because of this. The benefits of numerical methodology are so considerable that it will be used even if it cannot be justified rigorously. For the analyst this means not rejecting pragmatic approaches, but to come up with more effective ones. This is virtually terra incognita in numerical analysis. To begin exploring it we might try to classify pragmatics, examining their strength, when they are likely to succeed and under what conditions they can fail. If mathematics cannot provide rigorous error bounds, perhaps it can get us some way of assigning credibility to the computed results. This, I believe, is a very deep question. The machinery for investigating it may exist in the theory of probability or in the notion of fuzzy sets, but I know of no concerted effort addressing this problem. Yet, it is a very important issue; it would greatly benefit many people if we were to understand it a little better.

5. Producing information instead of numbers. Finally, I would like to mention one more point that deserves attention. What I have done so far is typical, but rather incomplete. I have used $\|\cdot\|$ without saying exactly what norm is meant. This has the advantage of generality, but the exact nature of the norm has to be specified before the result can be put to any practical use. The choice of the topology is important. Analysts tend to use the topology which makes the arguments simple; most often this means an inner product space, such as L_2 or a Sobolev-type space. Some finite-element analysis goes a step further and uses the *energy inner product*

$$\langle x, x\rangle_L = \langle Lx, x\rangle$$

which is convenient and suitable for positive-definite L.

Do we then have complete freedom to choose the topology and take whatever norm we like? The answer is clearly no, as we can illustrate by manipulating the problem to an absurd conclusion. As pointed out, the major difficulty with the a posteriori analysis is to put a good bound on $\|L^{-1}\|$; if I could always do this, I would have a universal method for bounding the accuracy of numerical computations. Of course, I cannot do this in general, but if you allow me enough freedom I can swindle the problem away altogether. Let me introduce the $f(udge)$-norm, defined by

$$\|x\|_f = \sqrt{\langle Lx, Lx\rangle}.$$

If L^{-1} is bounded, this is theoretically permissible. But then

$$\|x - x_n\|_f = \|\rho_n\|,$$

where the norm on the right might be the L_2 norm. Since $\|\rho_n\|$ is computable even for very complicated equations, I have solved the question of rigorous and computable error bounds for a large class of problems.

Life is unfortunately not so simple that we can solve difficult problems by a mere change of view. The sleight of hand here is too obvious to be acceptable. But then what is a permissible setting? Should we reject the f-norm, but accept the L_2 or energy norms? Or should we insist on a stronger maximum norm? These questions can be answered only by considering what we expect to get from our numerical computations.

What information we want from solving (1) must be taken into account by the analysis. If x is some function on R^n whose overall shape (say as it appears when plotted) is of interest, then an energy or L_2 norm analysis means very little. In such a situation the maximum norm ought to be used. On the other hand, if what we want is some quantity derived from x, a weaker norm may be justifiable. The latter is often the case. An engineer computing the flow field around a three-dimensional aerodynamic structure is rarely interested in that field to a high accuracy. His interest may be primarily in quantities derived by some sort of averaging, such as the lift and the drag on the structure. It could well be that in this case an L_2 analysis is sensible.

It is not too hard to give a preliminary formalism by which we can look at the question of what is and what is not useful information. Suppose there is a linear operator ϕ which associates the interesting information ϕx with the uninteresting solution x. In this case we will care little about $\|x - x_n\|$, but we will be concerned with bounding $\|\phi x - \phi x_n\|$. An a posteriori analysis trivially yields the inequality

$$\|\phi x - \phi x_n\| \leq \|\phi L^{-1}\| \, \|\rho_n\|. \tag{10}$$

We are now faced with having to get a bound, or at least a realistic estimate, for $\|\phi L^{-1}\|$. This may be easy or difficult, depending on the nature of the problem and the information we wish to obtain. But at least the a posteriori analysis provides a conceptually simple starting point for the questions that we should be asking.

All of this brings up a related issue. Computational mathematics may have been affected negatively by an overly close adherence to the paradigms of classical analysis. Classical analysis concentrates its efforts on getting a closed form solution of (1); the physically interesting answers are then derived from this. Numerical methodology has followed this lead, replacing the quest for a closed form solution by one for a numerical approximation. After an approximate solution x_n has been found, further numerical computations are needed to get ϕx_n. Since the relation between x_n and ϕx_n may not be clear to the user, there is a tendency to overcompute. The result is that many hours of computer time are wasted calculating to several significant digits, things that nobody wants to know. Once in a while one sees a comment on this; that instead of starting from (1), we ought to find formulations whose solution leads to ϕx_n directly. This is an intriguing suggestion that is worth pursuing but about which we know very little at the moment.

6. Conclusions. If we look at the present situation in computational mathematics, we find a wide range of success as well a some areas of failure, and we see that analysis has been the primary factor that determines the successes as well as the limitations. While practitioners rarely worry about mathematical rigor, the strength of the methodology that is used is very much dependent on what we know about the theory. Algorithm selection, which is supported by a well-understood convergence theory, can be done effectively by an unsophisticated user even for very complex problems. On the other hand, there is very little theory by which the numbers we get can be judged. The results that come out of the computer are therefore always suspect and sometimes downright wrong. It is clear that if any progress is to be made in this direction, a better theoretical foundation will have to be established. This is a task which, because of recent advances in computers and computer science, is becoming increasingly more critical.

Those who do numerical work are likely to agree that numerical methodology is a powerful but at the same time clumsy tool. A scientist who needs to solve a complicated partial differential equation numerically must usually do a large amount of tedious work. The equations have to be discretized first, then a computer program written and tested, after which numerical results are produced that must be further analyzed to get the desired information. The process is essentially straightforward, but takes a lot of time and it is very error-prone. Few would use it if there were any viable alternatives. Actually, the problem is about to get worse. With the development of computers capable of performing billions of arithmetic operations per second, scientists are beginning to tackle some extremely complex and only partially understood problems, such as three-dimensional turbulent fluid flow or atmospheric models for reliable weather prediction. These will have to be solved numerically in an environment of computer hardware and software more complicated by orders of magnitude than what we presently have. In such a situation, the answers will be very difficult to verify and many undetected errors will occur if the current casual approach continues to be used.

What is needed is not hard to state: a system which allows the user to describe problems in familiar mathematical notation, which can make an intelligent selection of a good algorithm, and which produces guaranteed results in the way the user wants them. Computer scientists have long strived to achieve this goal, but it is only now that it is within reach. A variety of systems, such as ELLPACK [**6**] for the solution of linear elliptic partial differential equations, are beginning to make their appearance. These systems are still fairly primitive and we can expect better ones as time goes on. Given what is currently happening, we can make some reasonable predictions. Future systems will not use numerical methodology exclusively, but will be able to solve problems by an effective mix of symbolic and numeric computations. These systems will have built into them a fair amount of rigorous mathematics and good pragmatics, kinds of "expert systems" that will outperform any but the most experienced numerical anslysts. Finally, the systems will converse with the user interactively so that they can be guided by the user's insight and physical intuition. It may take some time before such an ideal is realized, but the work has begun. When we start looking at such new ways of computing, we quickly realize that the current state of numerical analysis cannot support such systems adequately. As their development proceeds, the open problems I have outlined here will take on added significance. The automatic problem-solving systems of the future will need to be based on a strong theoretical foundation, so that they can produce results that are either guaranteed or have a high reliability as measured by some well-defined criteria. Otherwise, they will be viewed with suspicion by some and used incorrectly by others.

The advances that are now being made in various areas of computer science are forcing us to re-evaluate the way in which we do numerical computations. The realization that a different view is necessary has already created a new catch-word, *scientific computation*, which is becoming recognized as a separate discipline transcending the old computational mathematics. Mathematicians can play a significant role in this emerging discipline if they are able to provide new paradigms for computation, but in order to do so much of what has been done up to now will have to be reconsidered. There are some signs that such a fundamental rethinking is beginning. The work of Traub (e.g. [**8**]) and Smale [**7**] introduces ideas from information theory and computational complexity in an effort to understand more clearly what we can expect from numerical algorithms. My own work has convinced me that the a posteriori method is an idea with potential for major advances in making analysis practically useful, not only for the typical problems in differential equations, but also for the more difficult inverse and ill-posed problems [**3**]. But all of this work is in its early stages and has not yet had much impact on accepted numerical methodology. Perhaps the only importance of this activity is just to stir things up; eventually newer ideas may come along that will replace or subsume the older ones. What is exciting is that numerical analysts are beginning to look at their field in a very different way that might eventually lead to new and more powerful concepts. One hopes that these new concepts will not only answer unresolved practical problems, but will also make computational mathematics a richer and more attractive subject. Because introductory numerical analysis courses often emphasize a well-developed but uncritical methodology, the subject strikes many mathematicians as just a collection of "cook-book" formulas. This is unfortunate and discourages some to look at numerical analysis as an area for exciting research. But, as I tried to indicate, there are a large number of challenging problems in which mathematicians can make some very significant contributions. In fact, the full realization of the potential of computational mathematics requires the development of new and sophisticated analytical tools.

References

1. I. Babuska and W. C. Rheinboldt, *A-posteriori error estimates for the finite element method*, Internat. J. Numer. Methods Engrg. **12** (1978), 1597–1615.
2. E. Isaacson and H. B. Keller, *Analysis of numerical methods*, Wiley, New York,1966.
3. P. Linz, *Uncertainty in the solution of linear operator equations*, BIT **24** (1984), 92–101.
4. _____, *Precise bounds for inverses of integral equation operators*, Internat. J. Comput. Math. **24** (1988), 73–81.
5. _____, *Approximate solution of Fredholm integral equations with accurate and computable error bounds*, Tech. Report CSE-87-6, Division of Computer Science, Univ. of California, Davis, 1987.
6. J. R. Rice and R. F. Boisvert, *Solving elliptic problems using ELLPACK*, Springer-Verlag, Berlin and New York, 1985.
7. S. Smale, *Efficiency of algorithms of analysis*, Bull. Amer. Math. Soc. (N.S.) **13** (1985), 94–118.
8. J. Traub and H. Wozniakowsi, *A general theory of optimal algorithm*, Academic Press, New York, 1980.
9. O. C. Zienkiewicz and A. W. Craig, *A-posteriori error estimation and adaptive mesh refinement in the finite element method*, The Mathematical Basis of Finite Element Methods (D. F. Griffiths, ed.), Clarendon Press, 1984.

Division of Computer Science, University of California, Davis, California 95616