# SOURCES OF ERROR IN COMPUTATION

R.P. Brent

Computer Centre, Australian National University

## §1. INTRODUCTION

In the numerical approximation or prediction of a physical situation, errors may arise in the following ways:

1. There may be errors or oversimplifications in the formulation of the mathematical *model*.

2. To make a computational solution possible the model may have to be discretized. For example, integrals may be approximated by finite sums, and infinite series may be approximated by finite series. Thus *truncation* errors may be introduced.

3. There may be errors in the *data*. Such errors are usually unavoidable if the data are obtained from physical measurements or preliminary computations.

4. *Rounding* errors may occur during the computation.

Although model and truncation errors are extremely important, they are highly dependent on the problem, so we shall not discuss them here. In §2 we study the effect of errors in the data.

With numerically unstable methods rounding errors may be amplified by "catastrophic cancellation" (§3). With numerically stable methods this cannot occur, but many small errors may accumulate (§4). The amount of accumulation depends on the number system used, and different number systems are compared in §5.

We can only attempt a brief introduction to the subject of computational errors here. For further material, the reader is referred to Brent [1], Forsythe [2], Wilkinson [3], and the bibliographies given there.

## §2. THE CONDITION OF A PROBLEM

Without attempting to be precise, we say that a problem is *ill-conditioned* if small perturbations in the data cause large perturbations in the answer. Otherwise the problem is *well-conditioned*. There are several reasons for trying to avoid ill-conditioned problems:

1. Ill-conditioning is often a symptom of asking the wrong questions or formulating the problem incorrectly (see below).

2. If the data have moderately small errors, the answer may be completely wrong.

3. Even if the data are exact and a stable method is used, rounding errors in the computation may have an effect which is equivalent to perturbing the data slightly and then using exact arithmetic. Thus, the computed answer may be wrong.

To illustrate point 1, consider the system of simultaneous differential equations

$$d\underset{\sim}{x}/dt = B\underset{\sim}{x} ,$$

which has solutions of the form

$$\underset{\sim}{x} = \underset{\sim}{a}e^{\lambda t} ,$$

where $\lambda$ satisfies the polynomial equation

$$\det(\lambda I - B) = 0.$$

It is *not* desirable to compute the coefficients of this polynomial explicitly and then solve for $\lambda$, for the problem of determining $\lambda$ from the coefficients of the characteristic polynomial may be much worse conditioned than the original problem of determining $\underline{x}(t)$ (or of determining $\lambda$ from B). This is shown by the following example of Wilkinson [3, p.41], which also illustrates point 3 above.

Let B be the diagonal matrix with diagonal elements 1, 2, 3, ..., 20. Thus $\lambda$ satisfies the polynomial equation

$$p(\lambda) = 0 ,$$

where

$$p(x) = \prod_{i=1}^{20} (x - i)$$

$$= \sum_{j=0}^{20} a_j x^j \quad \text{say.}$$

Even if the coefficients $a_j$ are computed exactly, rounding errors in finding a zero of p by the customary methods will have the same effect as perturbing the $a_j$ slightly. For simplicity, suppose that only $a_{19}$ is perturbed. Let

$$q(x) = p(x) - 2^{-23}x^{19}$$

be the polynomial obtained by making a small perturbation $(-2^{-23})$ in $a_{19}$. Wilkinson [3, p.43] gives the zeros of q. The small ones are close to those of p, but the large ones are not at all close. For example, the zeros 18 and 19 of p become 19.50244 ± 1.94033i.

## §3. CATASTROPHIC CANCELLATION

We say that a numerical method for solving a problem is *stable* if it gives the exact solution to a slightly perturbed problem. A symptom of an unstable method is subtraction of two nearly equal numbers which have been contaminated with rounding errors. The subtraction removes the significant digits, leaving only rounding errors! D.H. Lehmer termed this *catastrophic cancellation*. Note

that the subtraction need not introduce any more rounding errors; it merely amplifies the effect of previous rounding errors.

For example, Forsythe [2, p.935] considers the computation of $e^{-5.5}$ from the expansion

$$e^x = \sum_{k=0}^{\infty} x^k/k! \ ,$$

using 5-decimal floating-point arithmetic. We obtain

$$\begin{aligned} e^{-5.5} &\approx 1.0000 - 5.5000 + 15.125 - 27.730 + 38.129 \\ &\quad - 41.942 + 38.446 - 30.208 + 20.768 - \ldots \\ &\approx 0.0026363 \ , \end{aligned}$$

but the correct answer is 0.0040868. Clearly, use of the Taylor series for computing $e^x$ is not stable for all real x. A stable method is to use

$$e^x = \left\{ \begin{array}{ll} \displaystyle\sum_{k=0}^{\infty} x^k/k! & \text{if } x \geq 0 \ , \\ \left( \displaystyle\sum_{k=0}^{\infty} (-x)^k/k! \right)^{-1} & \text{if } x < 0 \ . \end{array} \right\}$$

## §4. ACCUMULATION OF ERRORS

Even with stable methods, the effect of a large number of rounding errors may accumulate. This is usually much less serious than the effect of one rounding error amplified by catastrophic cancellation with an unstable method, but the accumulated errors may still be appreciable if a large number of arithmetic operations is performed.

For example, consider forming the product $x_0 x_1 \ldots x_n$ using fixed-precision floating-point arithmetic. We shall ignore the possibility of underflow or overflow. Let $\epsilon$ be the maximum error possible in forming one product, i.e.

$$\epsilon = \max \left| \frac{fl(a \times b) - ab}{ab} \right| \ ,$$

where the maximum is taken over nonzero floating-point numbers a and b such that

the computed product fl(axb) does not overflow or underflow. ($\varepsilon$ depends on the characteristics of the floating-point arithmetic and number representation.)

If $p_0 = x_0$ and

$$p_i = fl(p_{i-1} \times x_i)$$

$$= p_{i-1} x_i (1 - \delta_i) \quad \text{say,}$$

where

$$| \delta_i | \leq \varepsilon , \qquad i = 1, \ldots , n,$$

then the relative error in the final result is

$$\Delta = \frac{x_0 \ldots x_n - p_n}{x_0 \ldots x_n}$$

$$= 1 - \prod_{i=1}^{n} (1 - \delta_i)$$

$$= \sum_{i=1}^{n} \delta_i + \ldots$$

Thus, neglecting terms of order $n^2\varepsilon^2$, we have

$$| \Delta | \leq n\varepsilon .$$

This bound is rather pessimistic, for the errors $\delta_i$ may cancel rather than reinforce each other. Often $|\Delta|$ is of order $n^{\frac{1}{2}}\varepsilon$.

## §5. DIFFERENT NUMBER SYSTEMS

Suppose that we have a computer with a fixed word-length w bits and floating-point numbers with a fixed range

$$R = \log_2(f_{max}/f_{min}) ,$$

where the floating-point numbers are

$$-f_{max} < \ldots < -f_{min} < 0 < f_{min} < \ldots < f_{max} .$$

Various number systems are possible. For a normal floating-point system with base $\beta = 2^k$ and t digits, the constraint that the numbers (exponent and fraction) must fit into a w-bit word gives

$$2^{-kt_k} \geq 2^{1-w_R}$$

(see Brent [1]).

We may predict theoretically that the best choice is $\beta = 4$ (unless only normalized numbers are used and the leading fraction bit is implicit with $\beta = 2$). This has been verified empirically by Brent [1]. The effect of different choices of number system is illustrated by the following example (taken from [1]).

Let

$$\alpha_j = \left[ \sum_{i=1}^{n} (\lambda_i - \lambda_i^{(j)})^2 \right]^{\frac{1}{2}} \Big/ \|A\|_E ,$$

where A is a random n by n symmetric matrix whose eigenvalues are $\lambda_i$ and whose eigenvalues computed (by Householder's reduction to tridiagonal form and the QR algorithm) using number system j are $\lambda_i^{(j)}$, i = 1, ... , n. Let $\beta_j$ be the RMS value of $\alpha_j$ after 1000m trials.

Experimental results for several number systems are given in Table 1 (for further details see [1]). The experimental results agree fairly well with the theoretical predictions made in [1]. We may conclude that a computer with optimal floating-point hardware characteristics should have:

1. Unbiased rounded arithmetic (often more than twice as accurate as truncated arithmetic, and sometimes better by a factor of order $n^{\frac{1}{2}}$).
2. Base 2 (with the first bit of the fraction implicit) or base 4.
3. Several guard digits.

Unfortunately, for reasons of economy and speed in the arithmetic unit, few modern computers satisfy the above three requirements. For example, the results in Table 1 (and the other results of [1]) show that IBM's System/360 wastes about one decimal place of accuracy. Thus, double-precision computations are necessary more often than they should be.

| n | m | $\beta_2/\beta_1$ | $\beta_3/\beta_1$ | $\beta_4/\beta_1$ | $\beta_4'/\beta_1$ | $\beta_5/\beta_1$ |
|---|---|---|---|---|---|---|
| 2 | 100 | 1.50 | 2.00 | 2.23 | 5.67 | 14.2 |
| 4 | 10 | 1.68 | 2.00 | 2.71 | 7.90 | 19.5 |
| 8 | 3 | 1.76 | 2.05 | 3.27 | 9.48 | 26.0 |
| 16 | 1 | 1.82 | 1.99 | 3.49 | 10.7 | 28.8 |

Table 1: Comparison of different number systems*

* The number systems compared are:

1. $\beta = 2$, first bit implicit, rounded arithmetic, $\infty$ guard digits;
2. $\beta = 4$, first bit explicit, rounded arithmetic, $\infty$ guard digits;
3. $\beta = 2$, first bit explicit, rounded arithmetic, $\infty$ guard digits;
4. $\beta = 16$, first bit explicit, rounded arithmetic, $\infty$ guard digits;
5. $\beta = 256$, first bit explicit, rounded arithmetic, $\infty$ guard digits;
4'. $\beta = 16$, first bit explicit, truncated arithmetic, $\infty$ guard digits.

## §6. REFERENCES

[1] R.P. Brent, On the precision attainable with various floating-point number systems, Report RC 3751, IBM T.J. Watson Research Center, Yorktown Heights, New York, 1972. To appear in IEEE Trans. Comp.

[2] G.E. Forsythe, Pitfalls in Computation, or why a Math Book isn't enough, Amer. Math. Monthly 77(1970), 931-956.

[3] J.H. Wilkinson, Rounding errors in algebraic processes, Prentice-Hall, Englewood Cliffs, New Jersey, 1963.