(4) E.g., J. Rocek and T.-Y. Peng, *J. Am. Chem. Soc.*, **99**, 7624 (1977); E. A. Moelwyn-Hughes, "The Chemical Statics and Kinetics of Solutions", Academic Press, New York, 1971, pp 124-133.
(5) H. B. Mark, Jr., and G. A. Rechnitz, "Kinetics in Analytical Chemistry", Interscience, New York, 1968, 339 pp.
(6) K. Humski, V. Sendijarević, and V. J. Shiner, Jr., *J. Am. Chem. Soc.*, **95**, 7722 (1973).
(7) C. J. Collins, *Adv. Phys. Org. Chem.*, **2**, 62-74 (1964).
(8) W. E. Roseveare, *J. Am. Chem. Soc.*, **53**, 1651-1661 (1931).
(9) J. Casanova, Jr., and E. R. Weaver, *J. Chem. Educ.*, **42**, 137-139 (1965).
(10) R. C. Williams and J. W. Taylor, *J. Chem. Educ.*, **47**, 129-133 (1970).
(11) K. B. Wiberg in *Tech. Chem.* (*N.Y.*), **6**, 746 (1974).
(12) T. R. Dickson, "The Computer and Chemistry", W. H. Freeman, New York, 1968, pp 160-161.
(13) L. J. Reed and E. J. Theriault, *J. Phys. Chem.*, **35**, 673-689, 950-971 (1931); C. L. Perrin, "Mathematics for Chemists", Wiley, New York, 1970, pp 162-3.
(14) D. F. DeTar, *J. Chem. Educ.*, **44**, 759-761 (1967); S. Wold, *Acta Chem. Scand.*, **21**, 1986 (1967); D. F. DeTar, ref 16 pp 139-142; *ibid*, Vol. 4, 1972, pp 88-95.
(15) P. Moore, *J. Chem. Soc.*, *Faraday Trans. 1*, **68**, 1890 (1972), another exponential program, but written in ALGOL, with no published listing and an unspecified number of statements.
(16) D. F. DeTar, "Computer Programs for Chemistry", Vol. 1, W. A. Benjamin, New York, 1968, FORTRAN programs LSG, pp 117-125, and LSKIN1, pp 126-173.
(17) J. A. Beutler, *Chem. Eng. Prog.*, **50**, 569 (1954).
(18) KORE picks (n + 1)/2 as the middle datum (integer division, neglecting any remainder).
(19) C. G. Swain, M. S. Swain, and P. F. Strong, following paper in this issue.
(20) An extreme example is a precisely determined published rate constant with an error of more than five powers of ten caused by a systematic error, cited by C. G. Swain and R. E. Davis, *J. Am. Chem. Soc.*, **82**, 5949 (1960), ref 5
(21) I. Szele and H. Zollinger, *J. Am. Chem. Soc.*, **100**, 2811 (1978).
(22) These kinetic data were obtained by ultraviolet spectrometry. *Cf.* C. G. Swain, J. E. Sheats, K. G. Harbison, D. G. Gorenstein, and R. J. Rogers, *J. Am. Chem. Soc.*, **97**, 783-800 (1975).
(23) P. Cress, P. Dirksen, and J. W. Graham, "Fortran IV with WATFOR and WATFIV," Prentice-Hall, Englewood Cliffs, N.J., 1970.

A Simple, Reliable, and Fast Monte Carlo Method for Estimating the Uncertainty in Any Computer-Calculated Quantity

C. GARDNER SWAIN* and MARGUERITE S. SWAIN

Department of Chemistry, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

PETER F. STRONG

Arthur D. Little, Inc., Cambridge, Massachusetts 02140

Received May 18, 1979

UNCERT is a programmable method for evaluating the uncertainty (standard deviation) of any computer-calculated quantity, which is often easier to apply and freer of questionable assumptions than conventional propagation-of-error analyses. It utilizes a Monte Carlo strategy of multiple random trials, i.e., recalculations from slightly and randomly altered input data. Fifty trials generally yield a standard deviation with a relative precision (standard deviation) better than ±10%, which is more than adequate for most purposes. The method is illustrated by its application as an addendum to the first-order rate constant program of the preceding paper (with L. F. Berg).

INTRODUCTION

This programmable analytical method, which we call "UNCERT", was designed to determine uncertainties (standard deviations) of any quantities that are calculated from input data subject to random experimental errors or noise. It clearly excels standard propagation-of-error analyses when calculated quantities are determined iteratively (by successive approximations) owing to lack of explicit analytical expressions, or when standard error analyses allowing for interdependence (covariance) of the calculated quantities become too complicated and time-consuming. UNCERT is therefore potentially useful in computer calculations in many fields of science, engineering, and management. Scarcely any quantity is worth calculating without also establishing its uncertainty with a relative precision of ±50% or better. UNCERT quickly gives uncertainties with ±10% precision.

UNCERT is simpler in program logic and required coding than other methods of error analysis. It involves no matrix inversion nor even any calculation of a partial derivative. Instead it employs Monte Carlo recalculations of all the final calculated quantities many times from slightly and randomly altered input data. Then a separate standard deviation for each calculated quantity is derived from its root-mean-square variation in these random trials. UNCERT avoids biases that occur in other methods of estimating errors because they neglect higher order interactions and sometimes even first-order interactions among the calculated quantities. UNCERT is thus an acronym for "**un**biased **c**alculation of **er**rors by **r**andom **t**rials".

Monte Carlo solutions have been used in many other kinds of problems,[1-5] but only very rarely for error analysis.[6] Most treatments of error analysis overlook this approach entirely.[7] However, the following unique advantages make the Monte Carlo method even more desirable in error analysis than in other applications. First, the number of trials does not increase with the number of unknowns, and coding and execution time per trial increase only as the first power of the number of unknowns (quantities or parameters calculated), whereas the number of derivatives that had to be calculated by previous methods increased with the square of the number of unknowns. Therefore, the Monte Carlo method is simpler and faster than other methods in error analyses involving many unknowns. Second, the coding used to obtain the uncertainties is essentially just the coding used to solve for these unknowns initially. This is especially important in nonlinear problems, where the special extra coding required by other methods of error analysis is often long and complicated. It is also true, for reasons explained below, that each trial of the Monte Carlo method is typically much faster than the initial calculation, and is sometimes faster by as much as a factor of 50. Therefore,

0095-2338/80/1620-0051$01.00/0 © 1980 American Chemical Society

computer execution time has never proved to be a significant problem with any of the many kinds of programs into which we have incorporated UNCERT.

TWO VERSIONS OF UNCERT

Faster Version (1). Version 1 alone is expected to be adequate in more than 90% of all applications. It should always be used first, because, if a standard deviation s_f is calculated to be small (<5% of the quantity or function f to which it refers), practically the same value is obtained as would result from version 2, but the time per trial is less and fewer trials are required for any desired precision (relative standard deviation of the standard deviation). If a standard deviation s_f is found to be large (>20% of f), one can then decide whether the extra accuracy available by rerunning it by version 2 is needed. Usually it is not.

In version 1 each input datum is randomly perturbed (raised or lowered) in each trial by addition or, with equal probability, subtraction of s_x, the standard deviation for all data (x) of that type. However, data with different units (or data obtained by different methods or different investigators) should be considered as data of different type (belonging to different series) and should not be pooled for this purpose. If replicates are lacking, the standard deviation for data of a given type may be taken as 0.707 times the standard deviation between observed and calculated values for such data, determined after the initial calculation. The 0.707 ($=1/\sqrt{2}$) factor assumes that observed and calculated values have equal errors, hence that errors of their differences are larger than errors of either by a factor of $\sqrt{2}$.

For each input datum in each trial we make a separate decision whether to add or subtract s_x, the uncertainty of that datum, according to whether the next number from a 0 to 99 999 999 uniform random number generator is $\geq$ or $<$50 000 000. This procedure is extremely fast because no multiplication or division is required. We use the new uniform random number generator (URN) described in the following paper[8] which itself is very fast because it involves no MOD or other function, nor any multiplication or division.

The change in each recalculated quantity from its optimum value obtained in the initial calculation is then squared and accumulated in each trial, and the root mean square (square root of the quotient of the total divided by number of trials) becomes s_f, the standard deviation due to random errors for that quantity f.

s_s, the standard deviation of the derived standard deviation s_f, depends on the number of trials n as shown in the equation:

$$s_s = s_f/\sqrt{2n} \qquad (1)$$

Equation 1 is deduced in the Appendix under "Precision of the Uncertainties". s_s could, of course, be obtained by repeating version 1 a large number of times with different input data perturbations (from different sets of random numbers) and examining the resulting distribution of s_f values obtained. We have done this in the past to check eq 1, but now it is sufficient to execute version 1 only once to obtain a single s_f for each quantity of interest, and then use eq 1, which is derived from theory but not a part of the UNCERT program, to given the uncertainty or standard deviation of each s_f that UNCERT has calculated. For 200 trials, eq 1 gives an s_s/s_f of ±5%, hence 95% confidence that the standard deviations calculated are within twice ±5%, or ±10%, of the true standard deviations that would be approached as a limit if the number of trials were increased toward infinity. With 50 trials, s_s/s_f is ±10%; i.e., precision worsens by only a factor of 2 when the number of trials is cut by a factor of 4. For most purposes, derived standard deviations based on only 50 trials are good enough.

For their most important use in determining the number of significant figures in the best values of the quantities obtained in the initial calculation, even an order of magnitude (±1000%) is sufficient. In several different kinds of problems, we have found through UNCERT that some of the quantities we calculated are uncertain by more than the values of these quantities, while others of the same type have standard deviations of about 5%. The fact that 50 trials determine the standard deviations only approximately is of little importance compared to recognition that these latter quantities are determined to about two significant figures whereas the former are hardly determined at all. We had no feeling for the relative reliability of these different quantities prior to this application of UNCERT.

A major saving in computer execution time is obtained in iterative version 1 calculations by alteration or perturbation of the input data not by $\pm s_x$ but by a small fraction, CLIP (e.g., 10^{-3}), times $\pm s_x$. The derived standard deviations s_f of the calculated quantities are all finally divided by CLIP to compensate. This technique is described in more detail in the section below entitled "Use of CLIP in Version 1" and is strongly recommended for all version 1 applications.

Reserve Version (2). If a standard deviation s_f calculated by version 1 exceeds 20% of f, the true value of s_f may be even larger and should be redetermined by version 2 if accuracy is critical. If version 1 gives an s_f/f of 200%, version 2 may then show that it is really three or four powers of ten. However, the theoretical derivation of s_s/s_f, the relative standard deviation of the standard deviation, in the Appendix, is no longer valid because errors are assumed to be small. If s_s must be determined in the face of large errors, one must repeat the version 2 calculation many times to establish s_s empirically.

The basic difference in version 2 is that each input datum in each trial is perturbed not by random addition or subtraction of a constant s_x (standard deviation of the datum) as in version 1 but instead by addition of a random variable from a normal (Gaussian) distribution of values having a mean of zero and a standard deviation equal to s_x.[8] Thus the simulated errors are distributed more naturally. However, version 2 is slower than version 1 because each perturbation of an input datum in each of the trials typically takes about 12 times as much execution time as the simple addition or subtraction of s_x used in version 1, and also because more trials are required.

CLIP should not be used in version 2 but only in small-error situations where s_f is a nearly linear function of s_x. Use of CLIP in version 2 with large errors yields the same underestimates of standard deviations that are derived faster by version 1. However, we estimate that version 1 will be quite adequate by itself for nearly all applications of UNCERT. For version 1, use of CLIP is strongly recommended.

DETAILS

Iterative Calculations. UNCERT is always repetitive in the sense that enough trials (recalculations) are always performed to obtain standard deviations s_f of the precision (s_s/s_f) desired. In nonlinear problems the initial calculation and each recalculation of f may also be coded as an iterative process; i.e., the calculations may each begin with an approximation for one or more of the calculated quantities and produce a better successive approximation in each iterative cycle until convergence is attained. The following discussion applies only if UNCERT is iterative in this sense.

Although the initial calculation of f began with random numbers or rough estimates as starting approximations, the multiple recalculations or trials required for the UNCERT error analysis all begin instead with converged values from the initial calculation as starting approximations. Since these are much better starting estimates, each trial requires fewer cycles and is therefore less time-consuming than the initial calculation.

Use of CLIP in Version 1. A novel technique is used in version 1 to reduce further the number of iterative cycles required in each trial for reconvergence by causing all reconverged values to be much closer to the final values in the initial calculation which are used as starting estimates for each trial. To decrease the adjustment required in each trial, we perturb the input data x by only a small fraction called "CLIP" (constant for lowering input perturbations, e.g., 10^{-3}) of the s_x amount indicated above, then finally divide all the derived standard deviations of the calculated quantities by CLIP to compensate. To be more specific, we deflate the additions to or subtractions from the input data to 0.1% of s_x or to 0.0707% of the standard deviation between calculated and observed data of that type by multiplication by CLIP. Then 3–10 iterative cycles generally suffice in each trial, even when 10–500 cycles were required for convergence in the initial calculation.

In other types of problems a still smaller value for CLIP (such as 10^{-4}, 10^{-5}, 10^{-6}) may be desirable and necessary to reduce the number of cycles in each trial to a much lower and economically tolerable value. However, how low CLIP can be without damaging the precision s_s/s_f depends on the precision of the computer used, the s_s/s_f wanted, and the strictness of the convergence criterion. In general, the higher the precision of the computer, the higher the s_s/s_f allowed, or the more demanding the convergence criterion, the smaller CLIP can be. The computer precision required is highly dependent on the type of problem. Some problems calling for answers precise to five significant figures require only eight-figure precision in the calculations; for the same final precision, other types of problems may unexpectedly require computer precision equivalent to 16 decimal digits to avoid serious round-off errors when taking small differences between large numbers. For the example below (calculation of first-order rate constants), the computer must have a precision of eight or more decimal places and the lowest universally acceptable value of CLIP is 10^{-3} if one uses 200 trials (for s_s/s_f = 0.05) and convergence of k to $10^{-8}k$. In our experience 10^{-3} is a good value for CLIP, and the best one to use first if results from trying different CLIP's are to be compared. To determine the lowest acceptable value of CLIP for any completely new kind of problem, one tries values successively lower by factors of 10 until the average number of cycles per trial falls below 3. At this point one first finds a discrepancy slightly greater than anticipated from eq 1 between a (any) s_f value and the corresponding s_f values calculated from previous choices of CLIP, that is two to three times greater than the s_s given by eq 1. One then adopts a CLIP ten times larger. If one could be satisfied with a standard deviation an order of magnitude larger or if the convergence criterion were ten times more precise, CLIP could be smaller by a factor of 10. However, it is futile to try to reduce the total number of cycles in the error analysis by tightening up the convergence criterion, because the tendency toward fewer cycles from the then allowed smaller CLIP is just offset by the tendency toward more cycles from the stricter convergence criterion.

Two precautions should be observed when applying UNCERT to iterative calculations. For reliable standard deviations, it is essential at the beginning of each trial to reinitialize the starting approximations (all parameters and other values that can affect the result) to their final values at the end of the initial calculation. It is also essential to have the same kind and degree of convergence in each trial as in the initial calculation. CLIP can generally be set small enough to reduce the average number of cycles per trial to between 3 and 4 even in types of problems where 500 cycles may be required in the initial calculation. However, it should not be set so low as to reduce the average number of cycles per trial below 3 cycles; otherwise the standard deviations are likely to be less than 0.95 or greater than 1.05 times true standard deviation owing to either insufficient precision of the computer or poor matching of the degree of convergence between initial calculation and subsequent trials.

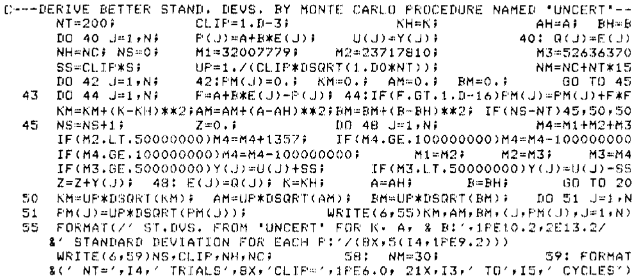
The most subtle threat to the precision of derived standard deviations arises when the convergence criterion differs qualitatively or quantitatively between the initial calculation and the subsequent trials. It is often desirable to employ overrelaxation factors to speed convergence in both the initial calculation and in the trials, as we do in the example below. It is then important that the overrelaxation factor used in the last cycle of each trial be the same as that used in the last cycle of the initial calculation. As a practical matter, it is simplest to make it the same for each cycle of each trial as that in the last cycle of the initial calculation. A more perplexing situation can arise in other problems where one may have also used occasional long extrapolations of one or more parameters in the initial calculation both to accelerate convergence and to provide a test for convergence through the smallness of changes in such extrapolations. If one foregoes extrapolations in the trials because they are so much shorter, what can one do to ensure the same degree of convergence as in the initial calculation? For a practical solution, one can further improve the convergence of the initial calculation after nominal convergence by continuing through two or more additional cycles (with an overrelaxation factor constant at its last value) and note the change in one parameter or in overall fit between the last two appended cycles. Then the same change (with the same overrelaxation factor) can be utilized as the criterion for convergence in each subsequent trial.

Because of these time-saving features, computer execution time has not been a serious or major problem in any of the applications that we have explored. For example, we found that machine execution time with 50 version 1 trials to obtain the separate standard deviations of each of 182 parameters obtained by a complicated nonlinear least-squares DOVE analysis[9] took only as much time as the initial calculation of the best values of these parameters (because reconvergence in each trial with a CLIP of 10^{-6} required only 4 cycles, whereas the initial calculation required 198 iterative cycles for convergence), and the total execution time for both on a Honeywell Multics 6180 time-sharing computer was only 44 s.

AN EXAMPLE: FIRST-ORDER RATE CONSTANTS

Coding. An application to illustrate the method is the calculation of errors (standard deviations) for first-order rate constants to higher precision than those given by the more approximate expressions built into the short program of the preceding paper.[10] To implement this improvement we need only add four arrays (P(50), Q(50), PM(50), and U(50)) to the DIMENSION statement, add an "IF(NM.GT.30)GO TO 27" statement after "IF(CK.LE.CN)GO TO 28;", add an "IF(NM.GT.30)GO TO 43" statement after "IF(CK.GT.CN)GO TO 10;", change 60 to 58 in the second statement after the one labeled 24, and add the new statements of Figure 1 between statements 39 and 60 of the short program. The Watfiv version of Fortran is used for reasons given previously.[10]

A number of trials NT sufficient for the precision desired (see "Faster Version (1)" above) is specified by the first arithmetic statement of Figure 1. CLIP is specified by the second statement. The next seven statements preserve final converged calculated values from the initial calculation, because these are the starting point for all trials (recalculations) and because the differences between these values and the



```
C---DERIVE BETTER STAND. DEVS. BY MONTE CARLO PROCEDURE NAMED 'UNCERT'--
      NT=200;            CLIP=1.D-3;                KH=K;           AH=A;  BH=B
      DO 40 J=1,N;       P(J)=A+B*E(J);     U(J)=Y(J);            40: Q(J)=E(J)
      NH=NC; NS=0;       M1=32007779;     M2=23717810;               M3=52636370
      SS=CLIF*S;         UP=1./(CLIP*DSQRT(1.D0*NT));                NM=NC+NT*15
      DO 42 J=1,N;       42:PM(J)=0.;  KM=0.;   AM=0.;   BM=0.;         GO TO 45
 43   DO 44 J=1,N;       F=A+B*E(J)-P(J); 44:IF(F.GT.1.D-16)PM(J)=PM(J)+F*F
      KM=KM+(K-KH)**2;AM=AM+(A-AH)**2;BM=BM+(B-BH)**2; IF(NS-NT)45,50,50
 45   NS=NS+1;           Z=0.;            DO 48 J=1,N;               M4=M1+M2+M3
      IF(M2.LT.50000000)M4=M4+1357;   IF(M4.GE.100000000)M4=M4-100000000
      IF(M4.GE.100000000)M4=M4-100000000;        M1=M2;     M2=M3;     M3=M4
      IF(M3.GE.50000000)Y(J)=U(J)+SS;      IF(M3.LT.50000000)Y(J)=U(J)-SS
      Z=Z+Y(J);   48: E(J)=Q(J); K=KH;      A=AH;        B=BH;          GO TO 20
 50   KM=UP*DSQRT(KM);   AM=UP*DSQRT(AM);   BM=UP*DSQRT(BM);     DO 51 J=1,N
 51   PM(J)=UP*DSQRT(PM(J));               WRITE(6,55)KM,AM,BM,(J,PM(J),J=1,N)
 55   FORMAT(/' ST.DVS. FROM 'UNCERT' FOR K, A, & B:',1PE10.2,2E13.2/
     &' STANDARD DEVIATION FOR EACH P:'/(8X,5(I4,1PE9.2)))
      WRITE(6,59)NS,CLIP,NH,NC;            58:  NM=30;               59: FORMAT
     &(' NT=',I4,' TRIALS',8X,'CLIP=',1PE6.0, 21X,I3,' TO',I5,' CYCLES')
```

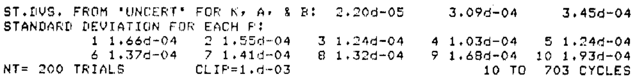
Figure 1. Additional statements needed to add UNCERT to the KORE program for first-order rate constants (Figure 1 of the preceding paper).

quantities obtained in the trials will be used to determine their standard deviations. In the present example there are 3 + N such quantities, namely K, A, B, and A + B*E(J) (predicted datum) for each of the N points (Y,T pairs) and these are preserved as KH, AH, BH, and P(J) for each point (J = 1,*N*). The measured datum Y(J) for each point is also preserved (as U(J)) because it must be randomly perturbed anew in each trial. The preservation of the exponentials E(J) as Q(J) in the present problem merely saves the time it would take to recalculate them. The other statements through 42 record the number of cycles at the beginning of UNCERT, zero the trial counter, specify three seeds for the random number generator, specify the fixed perturbation size (SS = CLIP*S) and the factor (UP) to be used at the end to compensate for CLIP and the number of trials (NT), specify an upper limit (NT*15) for the number of cycles in UNCERT, and zero the 3 + N variables (KM, AM, BM, PM(J)) destined to become the desired standard deviations.

For each trial, statements 45 through 48 increment the trial counter (NS), perturb each of the original input data (Y values) by either addition or subtraction of the fixed perturbation SS, depending on whether the next random number is as large as or smaller than the mean of its possible values (50 000 000), recalculate Z (the new sum of Y values) and recover the stored K, A, B, and E(J) values from the initial calculation. To provide a program compatible with all computers and all Fortran compilers, we have incorporated the uniform random number (URN) generator described in more detail in the following paper.[8] This appears to be quite satisfactory.

After convergence is attained in each trial, statements 43 through 44 cause the difference between new and initial values of A, B, K, and each P(J) to be squared and accumulated in one of 3 + N sums (AM, BM, KM, (PM(J), *J* = 1,N)). If the computer used can handle real numbers as low as 10^{-70}, as IBM 360 or 370 computers can, the IF(F.GT.1.D-16) condition on statement 44 can be deleted. Otherwise, it might be needed to avoid stops due to underflow if very small numbers were squared. After NT trials, statements 50 through 51 cause each of the 3 + N sums to be divided by NT, converted to a standard deviation by extraction of the square root, and multiplied by the reciprocal of CLIP, so that it becomes one of the desired standard deviations. The write statements record the results with suitable formats before return to the statement labeled 3 to look for new datasets to process.

For problems other than first-order rate constant calculations, it is, of course, necessary to reconsider what values are appropriate for NT and CLIP, to replace K, A, B, KH, AH, BH, P(J), KM, AM, BM, and PM(J) by other variables corresponding to the calculated quantities of interest, and correspondingly modify the write statements. However, these changes are easily made.



```
ST.DVS. FROM 'UNCERT' FOR K, A, & B:  2.20d-05     3.09d-04     3.45d-04
STANDARD DEVIATION FOR EACH P:
        1 1.66d-04   2 1.55d-04   3 1.24d-04   4 1.03d-04   5 1.24d-04
        6 1.37d-04   7 1.41d-04   8 1.32d-04   9 1.68d-04  10 1.93d-04
NT= 200 TRIALS        CLIP=1.d-03                        10 TO  703 CYCLES
```

Figure 2. Additional output after addition of UNCERT to KORE.

Results. In 200 trials UNCERT determines a standard deviation of 2.2×10^{-5} for the rate constant K (4.0413×10^{-2} s^{-1}) of the test dataset. This is better than the rough standard deviation (3.9×10^{-5}) calculated by the original program, and one can have 95% confidence that the calculated K does not deviate more than 2.1 times this s_k (i.e., more than 0.000 046) from its true value as a result of random errors. UNCERT also yields better standard deviations for A and B, and for each calculated datum P(J), each with the same relative reliability. Figure 2 displays the output using the input data of Table I of the preceding paper.

In this application, the less precise error analysis in the short program is usually adequate. Accordingly, we did not incorporate UNCERT into the program of the preceding paper.[10] However, we have employed this kind of error analysis in numerous analyses of data on chemical substituent effects and solvent effects, where it has proved invaluable for deciding whether or not differences in particular calculated parameters of the same type are outside of experimental error and therefore significant. These studies will be published separately.

APPENDIX

Precision of the Uncertainties (s_s/s_f). Equation 2 for the relative standard deviation of the estimated standard deviation for a simple random sample has been derived previously[11]

$$s_s/s_f \simeq [(\beta - 1)/4n]^{0.5} \tag{2}$$

where n is the number of f values (number of trials NT in UNCERT) and β is the kurtosis of the distribution of the quantity f for which s_f and s_s are evaluated.

The kurtosis of the distribution of a discrete variable f_i is defined by

$$\beta = \sum_i (f_i - \bar{f})^4/(ns_f^4) \tag{3}$$

where $\bar{f}$ is the mean of f_i values ($1 \leq i \leq n$). The kurtosis of any continuous function f is given by

$$\beta = (1/s_f^4) \int_{-\infty}^{+\infty} (f - \bar{f})^4 g(f)\, \mathrm{d}f \tag{4}$$

where $g(f)$ is the probability density function corresponding to any f. Distributions with $\beta < 3$ are called "platykurtic" (from Greek for broad arch); those with $\beta > 3$ are called "leptokurtic" (from Greek for thin arch). A two-line distribution (for a random variable that takes only two values with equal probability) has $\beta = 1.0$, the lowest possible kurtosis. Any rectangular (uniform or block) distribution yields 1.8 for β.[12,13] A normal (Gaussian) distribution gives 3. Because of the high (fourth) power of $f - \bar{f}$, β weights long tails more heavily than the standard deviation s_f does; hence β would be larger than 3 for a distribution that had relatively longer tails but a thinner peak than the normal distribution (like the kangaroos of the platypus–kangaroo mnemonic[14] (Figure 3).

The extreme two-line distribution exists in version 1 of UNCERT when the quantity or function f depends on only one input datum x, because the perturbations then take only two values (+CLIP*s_x or −CLIP*s_x) with equal probability, hence s_f is the same for every trial, $\beta = 1$ and $s_s = 0$. When only a *few* (2–4) input data affect f, values of β between 1.2 and 2.5 are common. When f depends on a great many input data, many small perturbations are combined and typically give a nearly normal distribution with a β of nearly 3. The normal distribution ($\beta = 3$) is expected to hold for random errors when there are an infinite number of small sources of error. In no application of UNCERT to any function f have

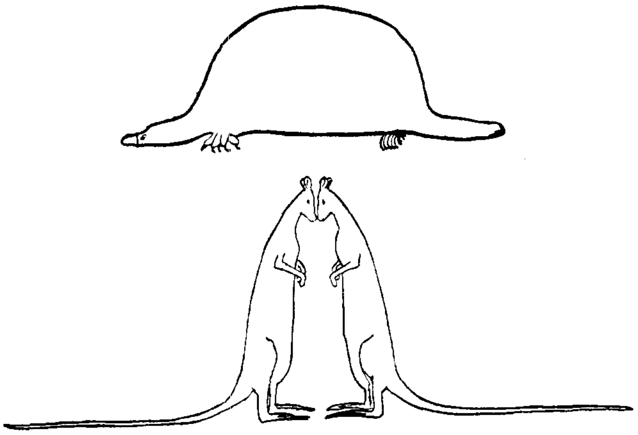


Figure 3. Platykurtic curves ($\beta < 3$) have short tails like a platypus (statistically his head is indistinguishable from his tail), while leptokurtic curves ($\beta > 3$) have long tails like kangaroos, noted for "lepping".[14] The distributions of f found in applications of UNCERT were all platykurtic.

we ever found β for f to exceed 3. We therefore consider $\beta = 3$ a practical upper limit and deduce from it and eq 2 that the practical estimate of s_s is that given by eq 5 or eq 1.

$$s_s/s_f = 1/\sqrt{2n} \qquad (5)$$

If only a few (2 to 4) input data (x) affect an f, and if $s_f < 0.05f$, then $s_s\sqrt{n}/s_f$ generally lies between 0 and $1/\sqrt{2}$. Typical values of $s_s\sqrt{n}/s_f$ from different functions f, determined by repeating version 1 many times with different random numbers, have ranged from 0.2 to 0.6. However, this small advantage over 0.71 does not warrant repeating version 1 to determine s_s empirically nor theoretically rederiving s_s for the number of data N, number of trials n, and specific function f involved. The practical and safe procedure is to use the number of trials n calculated from eq 5 for whatever s_s/s_f is desired.

The derivation of eq 5 is valid only if the errors are small enough to allow version 1 of UNCERT to be used.

Fallacy in a Simpler Estimate of s_s/s_f. It is noteworthy that an incorrect s_s/s_f would be deduced for normal distributions of f_i by the following propagation-or-variance analysis. The random variance of a function (w) of independently variable experimental quantities z_i is usually calculable from

$$s_w^2 = \sum_i \left[\left(\frac{\partial w}{\partial z_i}\right)^2 s_{z_i}^2\right] \qquad (6)$$

if the deviations of z_i are less than about 20% of their mean values.[15] However, if we were to apply this to the root-mean-square quantities that we use as standard deviations in version 1 (where $w = s_f$ and $z_i = f_i$), i.e., to eq 7

$$w = s_f = [\sum_i (f_i - f_0)^2/n]^{1/2} \qquad (7)$$

where f_0 is the quantity calculated in the initial calculation, f_i is calculated in a trial, and n is the number of trials, we would calculate the relative standard deviation of this standard deviation as

$$s_w/s_z = s_s/s_f = 1/\sqrt{n} \qquad (8)$$

This differs from eq 5 by a factor of $\sqrt{2}$. Experimentally, by repeating version 1 many times to obtain s_s empirically, we confirm that eq 5 is correct and eq 8 is incorrect.

The reason why eq 6 fails to yield eq 5 is failure of the implicit assumption underlying its use, namely that w is essentially linear in each z_i over the range of z_i values involved. That assumption does not hold in this application even if f_i varies less than 5%. Each partial derivative (eq 9) obtained by application of eq 6 to eq 7 has the form

$$\partial w/\partial z_i = \partial s_f/\partial f_i = (f_i - f_0)/(\sum_i (f_i - f_0)^2/n)^{1/2} \qquad (9)$$

Consider a particular i, e.g., $i = 5$ and the corresponding quantity (f_5) which is just one of the many possible f_i values. This f_i could by chance have exactly the value f_0, in which event the partial derivative of eq 9 would be zero, or could just as likely differ from f_0 by the root-mean-square amount, in which event this derivative would be unity. This derivative can thus vary by a factor of infinity and hence is not approximately constant over the actual range of f_i values. Therefore $s_f = w$ is not linear in each f_i ($=z_i$), and a necessary condition for validity of eq 6 is not satisfied.

This should serve as a warning against the indiscriminate use of eq 6 for estimating the standard deviation of a function from the standard deviations of its components, even when a simple analytical expression for the function exists. It is necessary to check, for each i, that w is a linear function of z_i, i.e., that $\partial w/\partial z_i$ is approximately constant over the actual range of variation of observed or calculated z_i values. This condition is not guaranteed by a very small (<5%) percentage variation in z_i values.

ACKNOWLEDGMENT

This work was supported in part by a research grant from the Donors of the Petroleum Research Fund administered by the American Chemical Society.

REFERENCES AND NOTES

(1) H. A. Meyer, Ed., "Symposium on Monte Carlo Methods", Wiley, New York, 1956.

(2) G. G. Lowry, Ed., "Markov Chains and Monte Carlo Calculations in Polymer Science", Marcel Dekker, New York, 1970.

(3) V. K. Smith, "Monte Carlo Methods", Lexington Books, Lexington, Mass., 1973. Application to econometrics.

(4) L. L. Carter and E. D. Cashwell, "Particle-Transport Simulation with the Monte Carlo Method", National Technical Information Service, U.S. Department of Commerce, Springfield, Va., 1975: 204 references.

(5) K. F. J. Heinrich, D. E. Newbury, and H. Yakowitz, Eds., "Use of Monte Carlo Calculations in Electron Probe Microanalysis and Scanning Electron Microscopy, U.S. National Bureau of Standards, Spec. Publ. 460, Washington, D.C., 1976.

(6) Y. Bard, "Nonlinear Parameter Estimation", Academic Press, New York, 1974, pp 46–47, 210–212.

(7) P. R. Bevington, "Data Reduction and Error Analysis for the Physical Sciences", McGraw-Hill, New York, 1969.

(8) C. G. Swain and M. S. Swain, following paper in this issue.

(9) P. F. Strong, C. G. Swain, and M. S. Swain, *J. Chem. Inf. Comput. Sci.*, **19**, 13 (1979).

(10) C. G. Swain, M. S. Swain, and L. F. Berg, preceding paper in this issue.

(11) M. H. Hansen, W. N. Hurwitz, and W. G. Madow, "Sample Survey Methods and Theory", Wiley, New York, 1953, pp 99–104. The exposition is adequate but the derivation of eq 2 is not simple.

(12) M. G. Bulmer, "Principles of Statistics", 2nd ed, M.I.T. Press, Cambridge, Mass., 1967, pp 63–65; H. M. Finucan, *J. R. Stat. Soc. B*, 111–112 (1964).

(13) Five other examples of symmetrical platykurtic distributions f, all having the same integrated area of 1.0, are (1) two rectangular bookends each 1.0 high and 0.5 wide, spaced 1.0 apart (variance = s_f^2 = 0.583, β = 1.14; (2) two L-shaped bookends, each with each side 1.0 long and 0.293 thick, placed 1.0 apart with tails out (s_f^2 = 0.811, β = 1.47); (3) the top half of the circle with radius $(2/\pi)^{1/2}$=0.798 (variance = $s_f^2 = 1/2\pi$ = 0.159, β = 2.00); (4) any isosceles triangle of unit area (β = 2.40); and (5) the top half of the four-pointed star having a span of $(2 + 2\sqrt{3})^{1/2}$ = 2.33754 between opposite points and 60° angles at the points (s_f^2 = 0.186, β = 2.85).

(14) "Student", *Biometrika*, **19**, 160 (1927); E. S. Pearson and J. Wishart, "Student's" Collected Papers', Biometrika Office, University College, London, 1943, p 145.

(15) Reference 7, p 60, eq 4–9; D. P. Shoemaker, C. W. Garland, and J. I. Steinfeld, "Experiments in Physical Chemistry" 3rd ed, McGraw-Hill, New York, 1974, p 52, eq 50.