1.0.1 Algorithms for computing validated results

Verification algorithms are somewhere on the boundary between computer algebra algorithms and numerical algorithms. With the former they share the strict validity of every result; like the latter they are very efficient, usually requiring a computing time only slower by a factor of five to ten compared to the best known numerical algorithm.

The basic principle of "verification algorithms" or "algorithms for computing validated results", also called "algorithms with automatic result verfication" is as follows. First, a pure floating point algorithm is used to compute an approximate solution for a given problem. This approximation is, hopefully, of good quality; however, no quality assumption is used at all. Second, a final verification step is appended. After this step, either error bounds are computed for the previously calculated approximation or, an error message signals that error bounds could not be computed. Any computed result is always perfectly correct in the sense that all possible conversion errors, rounding errors, approximation errors or others are rigorously estimated.

The calculation of the actual error bounds must use some estimation of the errors of the individual operations. This can be done by standard error analysis or, most convenient, by using interval operations. However, it is well known that extensive use of interval operations tends to weaken results. Therefore it is of utmost importance to diminish the overestimation due to successive use of interval operations.

This is the reason why the verification step is frequently based on a fixed point argument. Let the problem to find bounds for a solution of $f(x) = 0$ for some $f : \mathbb{R}^n \to \mathbb{R}^n$ be given. Then first the problem is transformed into a fixed point equation $g(x) = x$ with the property that the set of zeros of f and fixed points of g coincide.

The major advantage of interval arithmetic is the ability to estimate the *range* of a function, for example, the range of g over some n-dimensional interval vector X. This is done by simply replacing all operations by corresponding interval operations. The result, say Y, is definitive a superset of the true range $g(X)$. If the result Y is a subset of X, Brouwers Fixed Point Theorem ensures existence of a fixed point of g within Y, and henceforth existence of a zero of f within Y.

In other words, algorithms with result verification do verify the assumptions of mathematical theorems on the computer, where the assertion states a validated error bound for the solution.

Another advantage of this ability to estimate the range of a function over a certain domain arises in global search algorithms for zeros, like for example finding all (real or complex) zeros of a nonlinear function. The main problem of such algorithms - computer algebra, validated algorithms or others - is to *exclude* zeros from a certain region. This can be done by the cited range estimation, and sometimes in a very efficient way.

In the following we mention some standard problems for which validated algorithms are available with hints to the literature. General background on interval analysis, arithmetic and algorithms are in standard books, among them

[2, 5, 15, 16, 18].

For general systems of linear and nonlinear equations standard algorithms may be found in [12, 22, 23], for systems of equations with sparse matrix in [21] and in the literature cited over there. For global optimization algorithms and global search of all zeros of a function within a certain domain see [4, 5, 7, 8]. Ordinary systems of differential equations are treated in [14] and partial differential equations for example in [17, 19, 20]. For a good overview on current algorithms and methods see [6].

There are a number of public domain libraries for interval arithmetic and algorithms, where newer implementations include [9, 11]. Commercial libraries are available, among them [1, 3, 10, 13].

In summary, the objective of verification algorithms is to provide a certainty for pure numerical approximations by validated error bounds. They are designed to provide such error bounds whenever computation is possible within the limited floating point precision. The algorithms can be extended into never failing algorithms by increasing precision after temporary failure. In this case estimations of worst case computing times are possible using standard techniques from computer algebra. But, once again, this is not the main objective of validation algorithms.

However, the performance of computer algebra algorithms may be improved by simply applying a validation algorithm first. The additional computing time is usually small compared to the general algorithm, the results are still verified to be correct, and a verified result will be obtained if the problem is not too ill-conditioned with respect to the available floating point precision.

Another approach receives some attention recently. These are hybrid algorithms combining advantages of computer algebra and validated algorithms. A recent special issue of the Journal of Symbolic Computation (Number 6, December 1997) is devoted to this kind of algorithms.

Siegfried M. Rump (Hamburg)

References

1. ACRITH High-Accuracy Arithmetic Subroutine Library, Program Description and User's Guide. *IBM Publications*, (SC 33-6164-3), 1986.
2. G. Alefeld and J. Herzberger. *Introduction to Interval Computations.* Academic Press, New York, 1983.
3. ARITHMOS, Benutzerhandbuch. Siemens AG, Bibl.-Nr. U 2900-I-Z87-1, 1986.
4. T. Csendes and J. Pintér. The Impact of Accelerating Tools on the Interval Subdivision Algorithm for Global Optimization. *European Journal of Operational Research*, 65:314–320, 1993.
5. E.R. Hansen. *Global Optimization using Interval Analysis.* Marcel Dekker, New York, Basel, Hong Kong, 1992.
6. J. Herzberger, editor. *Topics in Validated Computations — Studies in Computational Mathematics.* Elsevier, Amsterdam, 1994.
7. C. Jansson. On Self-Validating Methods for Optimization Problems. In J. Herzberger, editor, *Topics in Validated Computations — Studies in Computational Mathematics 5*, pages 381–438, Amsterdam, 1994. North-Holland.

8. B. Kearfott and K. Du. The Cluster Problem in Global Optimization. *Computing Suppl.*, (9):117–127, 1993.
9. R.B. Kearfott, M. Dawande, K. Du, and C. Hu. INTLIB: A portable Fortran-77 interval standard function library. *ACM Trans. Math. Software*, 20:447–459, 1994.
10. R. Klatte, U. Kulisch, M. Neaga, D. Ratz, and Ch. Ullrich. *PASCAL-XSC — Sprachbeschreibung mit Beispielen.* Springer, 1991.
11. O. Knüppel. PROFIL / BIAS — A Fast Interval Library. *Computing 53*, pages 277–287, 1994.
12. R. Krawczyk. Newton-Algorithmen zur Bestimmung von Nullstellen mit Fehlerschranken. *Computing 4*, pages 187–201, 1969.
13. C. Lawo. C-XSC, a programming environment for verified scientific computing and numerical data processing. *In: Adams, E., Kulisch, U. (eds.) Scientific computing with automatic result verification: Academic Press*, pages 71–86, 1992.
14. R. Lohner. *Einschließung der Lösung gewöhnlicher Anfangs- und Randwertaufgaben und Anordnungen.* PhD thesis, University of Karlsruhe, 1988.
15. R.E. Moore. *Interval Analysis.* Prentice-Hall, Englewood Cliffs, N.J., 1966.
16. R.E. Moore. *Methods and Applications of Interval Analysis.* SIAM, Philadelphia, 1979.
17. M.R. Nakao. A Numerical Verification Method for the Existence of Weak Solutions for Nonlinear Boundary Value Problems. *Journal of Mathematical Analysis and Applications*, 164:489–507, 1992.
18. A. Neumaier. *Interval Methods for Systems of Equations, Encyclopedia of Mathematics and its Applications.* Cambridge University Press, 1990.
19. M. Plum. Computer Assisted Existence Proofs for two Point Boundary Value Problems. *Computing*, 46:19–34, 1991.
20. M. Plum. Numerical Existence Proofs and Explicit Bounds for Solutions of Nonlinear Elliptic Boundary Value Problems. *Computing*, 49:25–44, 1992.
21. S.M. Rump. Validated Solution of Large Linear Systems. In R. Albrecht, G. Alefeld, and H.J. Stetter, editors, *Computing Supplementum 9, Validation Numerics*, pages 191–212. Springer, 1993.
22. S.M. Rump. Verification Methods for Dense and Sparse Systems of Equations. In J. Herzberger, editor, *Topics in Validated Computations — Studies in Computational Mathematics*, pages 63–136, Elsevier, Amsterdam, 1994.
23. S.P. Shary. Optimal Solutions of Interval Linear Algebraic Systems. *Interval Computations 2*, pages 7–30, 1991.

This article was processed using the LaTeX macro package with LMAMULT style