Óbudai Egyetem

Doktori (PhD) értekezés tézisfüzete

Automatic roundoff error analysis of numerical algorithms

by

Attila Gáti

Témavezetők:

Prof. Dr. László Horváth

Prof. Dr. István Krómer

Alkalmazott Informatikai Doktori Iskola

Budapest, 2013

Contents

1 The background of the research

The idea of automatic error analysis of algorithms and/or arithmetic expressions is as old as the scientific computing itself and originates from Wilkinson (see Higham [27]), who also developed the theory of floating point error analysis [68], which is the basic of today's floating point arithmetic standards ([53], [50]). There are various forms of automatic error analysis with usually partial solutions to the problem (see, e.g. Higham [27],[21]). The most impressive approach is the interval analysis, although its use is limited by the technique itself (see. e.g. [47], [48], [49], [31], [32], [64], [27]). Most of the scientific algorithms however are implemented in floating point arithmetic and the users are interested in the numerical stability or robustness of these implementations. The theoretical investigations are usually difficult, require special knowledge and they not always result in conclusions that are acceptable in practice (for the reasons of this, see, e.g. [67], [68], [11], [27], [21]). The common approach is to make a thorough testing on selected test problems and to draw conclusions upon the test results (see, e.g. [57], [58], [65]). Clearly, these conclusions depend on the "lucky" selection of the test problems and in any case require a huge amount of extra work. The idea of some kind of automatization arose quite early.

The effective tool of linearizing the effects of roundoff errors and using the computational graph have been widely applied since the middle of the 70's ([40], [41], [42], [43], [44], [45], [46], [61], [35], [60]). Concerning the computational graph related error analysis Chaitin-Chatelin and Frayssé *[11]* gave the following summary.

"The stability analysis of an algorithm $x = G(y)$ with the implementation $G_{a\lg} = \Pi G^{(i)}$ depends on the partial derivatives $\frac{\partial G^{(i)}}{\partial y_j}$ computed at the various nodes of the computational graph (see § 2.6). The partial derivatives show how inaccuracies in data and rounding errors at each step are propagated through the computation. The analysis can be conducted in a forward (or bottom-up) mode or a backward (or top-down) mode. There have been several efforts to automate this derivation. One can cite the following:

1. Miller and Spooner (1978);

2. the B-analysis: Larson, Pasternak, and Wisniewski (1983), Larson and Sameh (1980);

3. the functional stability analysis: Rowan (1990);

4. the automatic differentiation (AD) techniques: Rall (1981), Griewank (1989), Iri (1991)"

Miller developed his approach in several papers ([37], [38], [39], [40], [41], [42], [43], [44], [45], [46]). The basic idea of Miller's method is the following. Given a numerical algorithm to analyze, a number $\omega(d)$ is associated with each set d

($d \in \mathbb{R}^n$) of input data. The function $\omega : \mathbb{R}^n \to \mathbb{R}$ measures rounding error, i.e., $\omega(d)$ is large exactly when the algorithm applied to d produces results which are excessively sensitive to rounding errors. A numerical maximizer is applied to search for large values of ω to provide information about the numerical properties of the algorithm. Finding a large value of ω can be interpreted as the given numerical algorithm is suffering from a specific kind of instability.

The software performs backward error analysis. The value $\omega(d) \cdot u$ (where u is the machine rounding unit) can be interpreted as the first order approximation of the upper bound for the backward error. The computation of the error measuring number is based on the partial derivatives of the output with respect to the input and the individual rounding errors. An automatic differentiation algorithm is used to provide the necessary derivatives.

The Miller algorithm was implemented in Fortran language (actually in FORTRAN IV) by Webb Miller and David Spooner [44], [45] in 1978. More information on the use of the software by Miller and its theoretical background can be found in [40], [41], [42] and [44]. The software is in the ACM TOMS library with serial number 532 [45].

Miller and Wrathall published a book [46] on the construction, background and use of the software in which the potential of the software is clearly demonstrated through 14 case studies. The answers of the software are consistent in these cases with the well known formal analytical and experimental results.

Miller's approach was further developed by Larson, Pasternak, and Wisniewski [35], Rowan [60], Bliss [5] and Bliss et al [6].

Upon the basis of corresponding literature, the Miller approach seems to be the most advanced although Miller's method has several setbacks. The numerical method to be analyzed must be expressed in a special, greatly simplified Fortran-like language. We can construct for-loops and if-tests that are based on the values of integer expressions. There is no way of conversion between real and integer types, and no mixed expressions (that contains both integer and real values) are allowed. Hence we can define only straight-line programs, i.e., where the flow of control does not depend on the values of floating point variables. To analyze methods with iterative loops and with branches on floating point values, the possible paths through any comparisons must be treated separately. This can be realized by constrained optimization. We confine search for maximum to those input vectors by which the required path of control is realized. The constraints can be specified through a user-defined subroutine.

Higham [27] points out that the special language and its restrictions as the greatest disadvantage of the software:

"*. . . yet the software has apparently not been widely used. This is probably largely due to the inability of the software to analyse algorithms expressed in For-*

tran, or any other standard language".

Unfortunately, this can be said more or less on the other developments that followed the Miller approach.

2 The aims of the research

The aim of my research work is to improve Miller's method to a level that meet today's requirements. I improved and upgraded Miller's method in two main steps. The first step was a F77 version usable in PC environment with standard F77 compilers such as the GNU and Watcom compilers. This version was able to handle algorithms with a maximum of 3000 inputs, and a maximum of 1000 outputs, and operations of a maximum of 50000 while the original Miller program can handle a maximum of 30 inputs, 20 outputs and 300 operations. However even this new version used the simplified Fortran like language of Miller which is considered as a major problem by Higham and others. Using the recently available techniques such as automatic differentiation, object oriented programming and the widespread use of MATLAB I have eliminated the above mentioned drawbacks of Miller method by creating a Matlab interface. Applying the operator overloading based implementation technique of automatic differentiation Griewank [25] and Bischof etal [4] we have provided means of analyzing numerical methods given in the form of Matlab m-functions. In our framework, we can define both straight-line programs and methods with iterative loops and arbitrary branches. Since the possible control paths are handled automatically, iterative methods and methods with pivoting techniques can also be analyzed in a convenient way. Miller originally used the direct search method of Rosenbrock for finding numerical instability. To improve the efficiency of maximizing, we added two more direct search methods [7]: the well known simplex method of Nelder and Mead, and the so called multidirectional search method developed by Torczon [63].

In the thesis we present a significantly improved and partially reconstructed Miller method by designing and developing a new Matlab package for automatic roundoff error analysis. Our software provides all the functionalities of the work by Miller and extends its applicability to such numerical algorithms that were complicated or even impossible to analyze with Miller's method before. Since the analyzed numerical algorithm can be given in the form of a Matlab m-file, our software is easy to use.

3 The methods of investigation

The investigations and software development required the knowledge and use of the following areas

- the theory of programming,
- theory of compilers,
- computational graphs,
- the theory and practice of automatic differentiation,
- backward error analysis theory of Wilkinson [67], [68] (see, also [27]),
- numerical analysis,
- nondifferentiable optimization methods
- object oriented programming.

4 The new scientific results

The numerical stability of computational algorithms is a very important issue. The classic error analysis techniques very rarely give computationally feasible results for the practitioner. The classical numerical testing on selected test examples often misleading depending on the selection and properties of the test problems. The purpose of the research is to provide an easy to use automatic error analysis based on compiler techniques, the computational graph techniques, the automatic differentiation techniques, object oriented programming. It was a definite aim that by simply using the written program of an algorithm under consideration the average or any other user can have a reliable estimate of the numerical stability without blind test problem selection. This line of research was initiated by W. Miller, who developed the most advanced program system and its theory during the 70s. Although many researcher developed similar or partly similar systems none of them achieved the high level of Miller's solution. Miller's solution however was limited in use by the computer technique of his age. In this thesis I analyzed, improved, upgraded and reimplemented his method. The results of the thesis can summarized as follows.

1. I replaced the minicompiler and its simplified programming language of the Miller method to object oriented Matlab.

2. Upon the bases of computer testing and theory I added two new optimization methods to the system that improved the performance of the software.

3. I reprogrammed and tested the system in Matlab. The new software provides all the functionalities of the work by Miller and extends its applicability to such numerical algorithms that were complicated or even impossible to analyze with Miller's method before. The analyzed numerical algorithm can be given in the form of a Matlab m-file. Hence our software is easy to use. The program consists of about 10000 lines and it is downloadable from the site

   $$\texttt{http}:\texttt{//phd.uni}-\texttt{obuda.hu/images/milleranalyzer.zip}$$

   together with a detailed user guide.

4. I applied the new Matlab version to investigate the numerical stability of some ABS methods (implicit LU, Huang and its variants), and three fast matrix multiplication algorithms. The obtained results indicate numerical instability of various scale and in the case of fast matrix multiplication algorithms give a definite yes for the suspected numerical instability of these methods.

5 The applicability of the results

In order to demonstrate the usability and efficiency of the developed software, we used it to examine the stability of some ABS methods [1], [2], namely the implicit LU methods and several variants of the Huang method [2]. The obtained computational results agreed with the already known facts about the numerical stability of the ABS algorithms. The program has shown that implicit LU is numerically unstable and that the modified Huang method has better stability properties than the original Huang method and the famous MGS (modified Gram-Schmidt) method. We also tested three famous fast matrix multiplication algorithms with the following results:

- The classical Winograd scalar product based matrix multiplication algorithm of $O(n^3)$ operation cost is highly unstable in accordance with the common belief, that has never been published.
- Both the Strassen and Winograd recursive matrix multiplication algorithms of $O(n^{2.81})$ operation costs are numerically unstable.
- The comparative testing indicates that the numerical stability of Strassen's algorithm is somewhat better than those of Winograd.

The obtained results support the common but disputed opinion that these fast matrix multiplication methods are numerically unstable (for reference, see, e.g. Higham [27]).

Upon the basis of our testing, we may think that the new software called Miller Analyzer for Matlab will be useful for numerical people or algorithm developers to analyze the effects of rounding errors.

6 List of references

[1] Abaffy, J., Broyden, C. G., Spedicato, E., A class of direct methods for linear systems, Numerische Mathematik, 45, 1984, 361–376

[2] Abaffy J., Spedicato, E.: ABS Projection Algorithms: Mathematical Techniques for Linear and Nonlinear Equations, Ellis Horwood, 1989

[3] Bini, D., Lotti, G.: Stability of fast algorithms for matrix multiplication, Numerische Mathematik, 36, 1980, 63–72

[4] Bischof, C.H., Bücker, H.M., Hovland, P., Naumann, U., Utke, J.(eds.): Advances in Automatic Differentiation, Springer, 2008

[5] Bliss, B.: Instrumentation of FORTRAN programs for automatic roundoff error analysis and performance evaluation, MSc thesis, University of Illinois at Urbana-Champaign, 1990

[6] Bliss, B., Brunet, M.-C., Gallopoulos, E.: Automatic parallel program instrumentation with applications in performance and error analysis, In Expert Systems for Scientific Computing, E. N. Houstis, J. R. Rice, and R. Vichnevetsky, editors, North-Holland, Amsterdam, The Netherlands, 1992, 235–260

[7] Brent, R.P.: Algorithms for matrix multiplication, Technical Report STAN-CS-70-157, Computer Science Department, Stanford University, 1970

[8] Brent, R.P.: Error analysis of algorithms for matrix multiplication and triangular decomposition using Winograd's identity, Numerische Mathematik, 16, 1970, 145–156

[9] Byatt, D.: Convergent variants of the Melder-Nead algorithm, MSc thesis, University of Canterbury, 2000

[10] Castano, B., Heintz, J., Llovet, J., Martinez, R.: On the data structure straight-line program and its implementation in symbolic computation, Mathematics and Computers in Simulation, Volume 51, Number 5, February 2000 , pp. 497-528(32)

[11] Chaitin-Chatelin, F., Frayssé, V.: Lectures on Finite Precision Computations, SIAM, Philadelphia, 1996

[12] Coleman, T.F., Verma, A.: ADMAT: An Automatic Differentiation Toolbox for MATLAB, Computer Science Department, Cornell University, 1998

[13] Conn, A.R., Scheinberg, K. Vicente, L.N.: Introduction to derivative-free optimization, SIAM, Philadelphia, 2008

[14] Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions, Journal of Symbolic Computation, 9, 1990, 251–280

[15] Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edition, The MIT Press, McGraw Hill, Cambridge, 2001

[16] D'Alberto, P., Nicolau, A.: Adaptive Strassen's matrix multiplication, In Proceedings of the 21st Annual International Conference on Supercomputing, ACM, New York, NY, 2007, 284–292.

[17] D'Alberto, P., Nicolau, A.: Adaptive Winograd's matrix multiplications, ACM Transactions on Mathematical Software, Vol. 36, No. 1, Article 3, DOI 10.1145/1486525.1486528 http://doi.acm.org/10.1145/1486525.1486528

[18] Demmel, J., Dumitriu, I., Holtz, O., Kleinberg, R.: Fast matrix multiplication is stable, Numerische Mathematik, 106, 2007, 199–224, DOI 10.1007/s00211-007-0061-6

[19] Douglas, C., , Heroux, M., Slishman, G., Smith, R.M.: GEMMW: A Portable Level 3 BLAS Winograd Variant of Strassen's Matrix-Matrix Multiply Algorithm, J. Comp. Phys., 110, 1994, 1–10

[20] Dumitrescu, B.: Improving and estimating the accuracy of Strassen's algorithm, Numerische Mathematik, 79, 1998, 485–499

[21] Einarsson, B. (ed): Accuracy and reliability in scientific computing, SIAM, Philadelphia, 2005

[22] Forth, S.A.: An efficient overloaded implementation of forward mode automatic differentiation in MATLAB, ACM Trans. Math. Softw., 32, 2006, 195–222

[23] Gao, F., Han, L.: Implementing the Nelder-Mead simplex algorithm with adaptive parameters, Comput. Optim. Appl., DOI 10.1007/s10589-010-9329-3

[24] Golub, G.H., Van Loan, C.F.: Matrix Computations, 2nd ed., Johns Hopkins University Press, 1989

[25] Griewank, A.: Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, SIAM, 2000

[26] Higham, N. J.: Exploiting Fast Matrix Multiplication Within the Level 3 BLAS, ACM Transactions on Mathematical Software, 16, 4, 1990, 352–368

[27] Higham, N. J.: Accuracy and Stability of Numerical Algorithms, SIAM, 1996

[28] Kaporin, I.: A practical algorithm for faster matrix multiplication, Numer. Linear Algebra Appl., 6, 1999, 687–700

[29] Kaporin, I.: The aggregation and cancellation techniques as a practical tool for faster matrix multiplication, Theoretical Computer Science 315, 2004, 469–510

[30] Kolda, T.G., Lewis, R.M., Torczon, V.: Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods, SIAM Review, 45, 2003, 385–482

[31] Krämer, W.: Constructive Error Analysis, Journal of Universal Computer Science, vol. 4, no. 2 (1998), 147-163

[32] Krämer, W., Bantle, A.: Automatic Forward Error Analysis for Floating Point Algorithms, Reliable Computing 7, 2001, 321–340

[33] Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P. E.: Convergence Properties of the Nelder–Mead Simplex Method in Low Dimensions, SIAM J. Optim., 9 (1999) 112–147

[34] Lakshmivaran, S., Dhall, S.K.: Analysis and Design of Parallel Algorithms, McGraw-Hill, 1990

[35] Larson, J.L., Pasternak, M.E.,Wisniewski, J.A.: Algorithm 594: Software for Relative Error Analysis, ACM Trans. Math. Softw., 9, 1983, 125–130

[36] Lewis, R.M., Torczon, V., Trosset, M.W.: Direct search methods: then and now, Journal of Computational and Applied Mathematics 124 (2000) 191–207

[37] Miller, W.: On the stability of finite numerical procedures, Numerische Mathematik, 19, 1972, 425–432

[38] Miller, W.: Computational complexity and numerical stability, in Proceeding STOC '74 Proceedings of the sixth annual ACM symposium on Theory of Computing, 1974, 317–322

[39] Miller, W.: Computational complexity and numerical stability, SIAM J. Comput., 4, 2, 1975, 97–107

[40] Miller, W.: Computer search for numerical instability, JACM, 22, 4, 1975, 512–521

[41] Miller, W.: Software for roundoff analysis, ACM Trans. Math. Softw., 1, 2, 1975, 108–128

[42] Miller, W.: Roundoff analysis by direct comparison of two algorithms, SIAM Journal on Numerical Analysis, 13, 3, 1976, 382–392

[43] Miller, W.: Roundoff Analysis and Sparse Data, Numerische Mathematik, 29, 1977, 37–44

[44] Miller, W., Spooner, D.: Software for roundoff analysis, II, ACM Trans. Math. Softw., 4, 4, 1978, 369–387

[45] Miller, W., Spooner, D.: Algorithm 532: software for roundoff analysis [Z], ACM Trans. Math. Softw., 4, 4, 1978, 388–390

[46] Miller, W., Wrathall, C.: Software for roundoff analysis of matrix algorithms, Academic Press, New York, 1980

[47] Monte Leon, V.J.: Automatic error analysis in finite digital computations, using range arithmetic, MSc thesis, U.S. Naval Postgraduate School, Monterey, California, 1966

[48] Moore, R.E.: Automatic Error Analysis in Digital Computation, Technical report, LMSD-48421, Lockheed, 28 January, 1959

[49] Moore, R.E.: Methods and Applications of Interval Analysis, SIAM, Philadelphia, 1979

[50] Muller, J.-M., et al.: Handbook of Floating-Point Arithmetic, Birkhäuser, 2010

[51] Mutrie, M.P.W., Bartels, R.H., Char, B.W.: An approach for floating-point error analysis using computer algebra, Proceeding ISSAC '92 Papers from the international symposium on Symbolic and algebraic computation, ACM New York, NY, USA, 1992, 284–293

[52] Nelder, J.A., Mead, R.: A simplex method for function minimization, Computer Journal, 7 (1965) 308–313

[53] Overton, M.L.: Numerical computing with IEEE floating point arithmetic, SIAM, Philadelphia, 2001

[54] Pan, V.: How can we speed up matrix multiplication?, SIAM Review, 26, 3, 1984, 393–415

[55] Rall. L.B.: Automatic Differentiation: Techniques and Applications, Lecture Notes in Computer Science, 120, Springer, 1981

[56] Register, A.H.: A Guide to MATLAB Object-Oriented Programming, Chapman & Hall/CRC, 2007

[57] Rice, J.R. : Numerical Methods, Software, and Analysis, McGraw-Hill, 1983

[58] Rice, J.R.: Matrix Computations and Mathematical Software, McGraw-Hill, 1983

[59] Rosenbrock, H.H.: An automatic method for finding the greatest or least value of a function, Comput. J., 3, 1960, 175–184

[60] Rowan, T.H.: Functional Stability Analysis of Numerical Algorithms. Ph.D. thesis, University of Texas at Austin, Austin, 1990

[61] Stoutemyer, D.R.: Automatic error analysis using computer algebraic manipulation. ACM Trans. Math. Software, 3, 1977, 26–43

[62] Strassen, V.: Gaussian elimination is not optimal, Numerische Mathematik, 13, 1969, 354–356

[63] Torczon, V.: On the convergence of the multidirectional search algorithm, SIAM Journal on Optimization, 1, 1, 1991, 123–145

[64] Tucker, W.: Validated numerics: a short introduction to rigorous computations, Princeton University Press, 2011

[65] Ueberhuber, C.W.: Numerical Computation 1-2 (Methods, Software, and Analysis), Springer, 1997

[66] Verma, A.: ADMAT: Automatic Differentiation in MATLAB Using Object Oriented Methods, in M. E. Henderson and C. R. Anderson and S. L. Lyons (eds.), Object Oriented Methods for Interoperable Scientific and Engineering Computing, Proceedings of the 1998 SIAM Workshop, SIAM, Philadelphia, 174–183, 1999

[67] Wilkinson, J.H.: The Algebraic Eigenvalue Problem, Oxford University Press, 1965

[68] Wilkinson, J.H.: Rounding Errors in Algebraic Processes, Dover, 1994

[69] Winograd, S.: A new algorithm for inner product, IEEE Trans. C-17, 1968, 693–694

[70] Winograd, S.: On multiplication of 2×2 matrices, Linear Algebra Appl., 4, 1971, 381–388

7 List of references related to the thesis

[1] Gáti Attila: Numerikus algoritmusok automatikus hibaanalízise Miller módszerével, Doktoranduszok Fóruma, Gépészmérnöki Kar Szekciókiadványa, Miskolci Egyetem 2003, 70–77

[2] Gáti, A.: Automatic error analysis with Miller's method. Miskolc Math. Notes, 5, 1, 2004, 25–32.

[3] Gáti, A.: Automatic roundoff error analysis with Miller's method, MicroCAD 2004, International Scientific Conference, Applied Mechanics, Modern Numerical Methods, University of Miskolc 2004, 29–34

[4] Gáti Attila: A Miller-Spooner-féle automatikus hibaelemző program továbbfejlesztése, Doktoranduszok Fóruma, Gépészmérnöki Kar Szekciókiadványa, Miskolci Egyetem 2004, 83–87

[5] Gáti, A.: The upgrading of the Miller-Spooner roundoff analyser software, MicroCAD 2005, International Scientific Conference, Applied Mechanics, Modern Numerical Methods, Miskolci Egyetem 2005, 49–54, ISBN 963 661 653 1

[6] Gáti, A.: The upgrading of the Miller-Spooner roundoff analyser software, 5th International Conference of PhD Students, Engineering Sciences II, Miskolci Egyetem 2005, 43–48. ISBN 963 661 679 5

[7] Gáti, A.: A Miller-Spooner-féle automatikus hibaelemző program továbbfejlesztése (The upgrading of the Miller-Spooner roundoff analyzer software), In: A. Pethő and M. Herdon (Eds.): Proceedings of IF 2005, Conference on Informatics in Higher Education, Debrecen, 2005, University of Debrecen, Faculty of Informatics (in Hungarian), ISBN 963 472 909 6

[8] Gáti Attila: Hibaelemzés és automatikus differenciálás, Doktoranduszok Fóruma, Gépészmérnöki Kar Szekciókiadványa, Miskolci Egyetem 2005, 56–62

[9] Gáti, A.: Roundoff error analysis and automatic differentiation, MicroCAD 2006, International Science Conference, Mathematics and Computer Science, Miskolci Egyetem 2006, 29–35. ISBN 963 661 707 4

[10] Gáti Attila: Hibaelemzés és automatikus differenciálás, Tavaszi Szél 2006 Konferenciakiadvány, Kaposvár 2006, 287–290. ISBN 963 229 773 3

[11] Gáti, A.: Miller Analyser for Matlab, User's Manual. Available on: http://phd.bmf.hu/images/milleranalyzer.zip

[12] Gáti, A.: Miller Analyzer for Matlab: A Matlab Package for Automatic Roundoff Analysis, Computing and Informatics, 31, 2012, 713–726