# Variable precision in mathematical and scientific computing
# Proposal for an ICERM workshop

April 10, 2019

## 1 Organizing committee

1. David H. Bailey, LBNL (retired) and UC Davis [Variable precision in mathematical computing; HPC applications]. Principal point of contact.
2. Neil Burgess, ARM, Cambridge, U.K. [Design and usage of floating-point hardware].
3. Jack Dongarra, Univ. of Tennessee [Variable precision in numerical linear algebra].
4. Alyson Fox, LLNL [Compression algorithms for floating-point data; HPC applications].
5. Jeffrey A. F. Hittinger, LLNL [Variable precision for HPC applications].
6. Cindy Rubio-González, UC Davis [Tools for variable precision, including "Precimonious"].

## 2 Background

From its introduction in the 1980s, the IEEE-754 standard for floating-point arithmetic has ably served a wide range of users: mathematicians, computer scientists, physicists, chemists, biologists, social scientists and engineers. William Kahan, who led the development of this standard, later received the ACM Turing Award for this work. The initial standard specified 32-bit ("single") and 64-bit ("double") floating-point arithmetic, which were quickly adopted by processor vendors. Even today, the vast majority of numerical computations in research and engineering employ either IEEE single or IEEE double, typically one or the other exclusively in a single application.

However, recent developments have exhibited the need for a broader range of precision levels, and variable precision within a single application. There are clear performance advantages to a variable precision framework: faster processing, better cache utilization, lower memory usage and lower long-term data storage. But effective usage of variable precision requires a more sophisticated mathematical framework, together with corresponding software tools and diagnostic facilities.

At the low end, the explosive rise of graphics, artificial intelligence and machine learning has underscored the utility of reduced precision levels. Accordingly, an IEEE 16-bit "half" precision standard has been specified, with five exponent bits and ten mantissa bits. Many in the machine learning community are using the "bfloat16" format, which has eight exponent bits and seven mantissa bits. Hardware such as NVIDIA's tensor core units can take advantage of these formats to significantly increase processing rates.

At the same time, researchers in the high-performance computing (HPC) field, in a drive to achieve exascale computing, are also reconsidering their usage of numerical precision, since as mentioned above there are clear performance advantages to employing reduced precision where possible. This has led to new mixed-precision approaches for common linear algebra operations [1, 2] and renewed interest in iterative refinement, where initial iterations are performed using half- or single-precision [8]. Along this line, recognizing that for many simulations much of the data stored in a IEEE 64-bit double precision variable has low information content (consider that many of the bits are approximation errors), researchers are exploring the use of lossy floating point compression, not only for I/O, but also for storing solution state variables [10, 5].

Exascale computing has also exposed the need for even greater precision than IEEE 64-bit double in some cases, because greatly magnified numerical sensitivities often mean that one can no longer be certain that results are numerically reliable. One remedy is to use IEEE 128-bit quad precision in selected portions of the computation, but to date 128-bit IEEE has not yet been implemented in hardware by the major microprocessor vendors. It is, however, now available via software in some compilers, notably the gfortran compiler, and so is being used in some research applications. As a single example, researchers at Stanford have had remarkable success in using quad precision in multiscale linear programming applications in biology [12].

There has also been a rise in the usage of very high precision (hundreds or even thousands of digits). For example, numerous new results have been discovered by computing mathematical expressions to very high precision, and then using integer relation algorithms such as the "PSLQ" algorithm to recognize these numerical values in terms of simple mathematical formulas. Among the results that have been discovered in this fashion are new formulas connecting mathematical constants [4] and the elucidation of polynomials connected to the Poisson potential function of mathematical physics (the latter requiring up to 64,000-digit precision) [3]. Such computations are most efficiently performed using a dynamically varying level of precision, doing as much computation as possible with standard precision and only invoking very high precision when necessary.

In summary, although the IEEE 754 floating-point standard has served the mathematical, scientific and engineering world very well for over 30 years, we now are seeing rapidly growing demand for reduced precision (machine learning, neural nets, graphics, etc.), a growing need for mixed 32-64-bit precision, and also a need for greater than 64-bit, all typically varying within a given application. To the extent that IEEE-754 fails to adequately meet new demands such as these, researchers are considering completely different alternatives, for which a flexible precision level is a fundamental feature of the design [7, 11], and are exploring new mathematical and software frameworks to better understand and utilize such facilities.

## 3 Research questions to be explored

To this end, here is a list of research questions in variable precision, encompassing both mathematics and computer science, that we propose to discuss in this workshop. By "variable precision" here, we mean "exploiting different precision levels seamlessly to maximize computational performance given available resources, while retaining numerical stability, integrity and reproducibility":

1. How are scientific disciplines, ranging from pure mathematics to applied science and engineering, currently using variable precision? What new opportunities are on the horizon?

2. What does the explosive growth in artificial intelligence and machine learning mean for hardware, software and tools to support variable precision? What are the fundamental accuracy requirements of such computations?

3. Are there new mathematical approaches to efficiently and accurately compute various mathematical operations and functions to both very modest precision (2-3 digit accuracy) and very

high precision (hundreds or thousands of digits)? What research has been done here, and what needs to be done?

4. What new formats and representation systems to facilitate variable level precision computing are available [7, 11]? To what extent do these new proposals overcome strengths and weaknesses in existing systems such as IEEE-754 hardware and MPFR software [6]?

5. What new support is needed from computer hardware and software vendors for variable precision computing? Could some modest hardware changes be made to support low, variable precision and high precision, rather than relying on all-software implementations?

6. Almost all of published research in numerical mathematics and numerical analysis to date has focused on either 32-bit or 64-bit results. Are there other techniques that may be superior in a variable precision environment?

7. Classical error bounds used in numerical analysis, such as those developed by Wilkinson, typically assume worst-case scenarios and thus are not very useful in a limited precision or variable precision environment. What research has being done or should be done to develop more realistic error estimates [9]?

8. How can iterative refinement best be performed in a variable precision environment? What new opportunities are there for using iterative refinement with variable precision?

9. What algorithms and software techniques can effectively compress floating-point data, on-the-fly, to save data storage and transfer? Can improved schemes be devised?

10. What software tools are available to automatically or semi-automatically analyze an application code, identify numerically sensitive spots and propose or implement remedies? What other tools are needed?

11. How can we ensure that computations with variable precision levels are reproducible? What does reproducibility even mean when computing, say, with only 16-bit precision?

## 4 Objective of proposed workshop and selection of participants

The organizers plan to invite approximately 40–50 persons from the computational mathematics, computer science, data science, applications and vendor communities to participate in a five-day discussion of state-of-the-art developments in variable precision computing and to assess the next steps needed in this arena. In particular, participants will be selected based on specialized expertise in one of the topic areas, their likely contribution to answering the research questions listed in the previous section, and their potential contribution to a report (see next section).

Participants will also be selected to cover a diverse range of geographical areas (both within U.S. and non-U.S.), as well as to promote gender and racial diversity. The proposed committee as shown above includes two women, one of which has a Hispanic background, and a combination of junior- and senior-level researchers. Each of these persons is very well qualified to lead and participate in this workshop. To further promote diversity, we also plan to include a two-hour event during the program specifically dedicated to presentations and networking for underrepresented groups.

## 5 Report

The organizing committee, working in collaboration with all participants in the meeting, will produce a report of the workshop's findings of what research has been done in this area and what research needs to be done moving forward. We plan to submit this report to funding agencies for future research and development in the field.

## References

[1] M. Baboulin, A. Buttari, J. Dongarra, J. Kurzak, J. Langou, J. Langou, P. Luszczek and S. Tomov, "Accelerating scientific computations with mixed precision algorithms", *Comp. Phys. Comm.*, vol. 180 (2009), 2526–2533, `http://dx.doi.org/10.1016/j.cpc.2008.11.005`.

[2] A. Buttari, J. Dongarra, J. Kurzak, P. Luszczek and S. Tomov, "Using mixed precision for sparse matrix computations to enhance the performance while achieving 64-bit accuracy," *ACM TOMS*, vol. 34 (2008), art. 17, `http://doi.acm.org/10.1145/1377596.1377597`.

[3] D. H. Bailey, J. M. Borwein, J. Kimberley and W. Ladd, "Computer discovery and analysis of large Poisson polynomials," *Experimental Mathematics*, vol. 26 (2016), 349–363, `https://www.tandfonline.com/doi/full/10.1080/10586458.2016.1180565`.

[4] D. H. Bailey, J. M. Borwein, A. Mattingly and G. Wightwick, "The computation of previously inaccessible digits of $\pi^2$ and Catalan's constant," *Notices of the AMS*, (2013) 60(7), 844–854, `http://www.ams.org/notices/201307/rnoti-p844.pdf`.

[5] J. Diffenderfer, A. Fox, J. Hittinger, G. Sanders, and P. Lindstrom, "Error analysis of ZFP compression for floating-point data," *arXiv e-prints*, 2018, `https://arxiv.org/abs/1805.00546`.

[6] L. Fousse, G. Hanrot, V. Lefevre, P. Pelissier and P. Zimmermann, "MPFR: A multiple-precision binary floating-point library with correct rounding," *INRIA Research report RR-5753*, Nov 2005, `https://hal.inria.fr/inria-00070266`.

[7] J. Gustafson and I. Yonemoto, "Beating floating point at its own game: Posit arithmetic," 2017, `http://www.johngustafson.net/pdfs/BeatingFloatingPoint.pdf`.

[8] A. Haidar, S. Tomov, J. Dongarra, and N. J. Higham, "Harnessing GPU tensor cores for fast FP16 arithmetic to speed up mixed-precision iterative refinement solvers," in *Proc. of SC18* (2018), IEEE Press, Piscataway, NJ, USA, Article 47, `https://doi.org/10.1145/3148226.3148237`.

[9] N. J. Higham and T. Mary, "A new approach to probabilistic rounding error analysis," *MIMS Preprint 2018.33*, 12 Nov 2018, `http://eprints.maths.manchester.ac.uk/2673/`.

[10] P. Lindstrom, "Fixed-Rate Compressed Floating-Point Arrays," *IEEE Trans. Vis. Comp. Graphics*, vol. 20 (2014), 2674–2683, Dec 2014, `https://doi.org/10.1109/TVCG.2014.2346458`.

[11] P. Lindstrom, S. Lloyd and J. Hittinger, "Universal coding of the reals: Alternatives to IEEE floating point," in *Proc. Next Gen. Arithmetic (CoNGA '18)*, (2018) ACM, New York, NY, USA, Article 5, `https://doi.org/10.1145/3190339.3190344`.

[12] D. Ma and M. Saunders, "Solving multiscale linear programs using the simplex method in quadruple precision," in M. Al-Baali, L. Grandinetti and A. Purnama, ed., *Recent Developments in Numerical Analysis and Optimization*, Springer, 2017, `http://web.stanford.edu/group/SOL/reports/quadLP3.pdf`.