ExBLAS: Reproducible and Accurate BLAS Library

Roman Iakymchuk, Stef Graillat, Sylvain Collange, and David Defour

Sorbonne Universités, UPMC Univ Paris 06, ICS, UMR 7606, LIP6, F-75005 Paris, France
INRIA – Centre de recherche Rennes – Bretagne Atlantique, F-35042 Rennes Cedex, France
DALI–LIRMM, Université de Perpignan, 52 avenue Paul Alduy, F-66860 Perpignan, France

Abstract

We aim at providing algorithms and implementations for fundamental linear algebra operations – like the ones included in the Basic Linear Algebra Subprograms (BLAS) library – that would deliver reproducible and accurate results with reasonable performance overhead compared to the standard non-reproducible implementations on modern parallel architectures such as Intel processors, Intel Xeon Phi co-processors, and GPU accelerators.

Introduction

In general, BLAS routines rely on the optimized version of parallel reduction and dot product involving floating-point additions and multiplications operations that are non-associative. Due to non-associativity of these operations and dynamic scheduling on parallel architectures, getting a bitwise reproducible floating-point result for multiple executions of the same code on different or even similar parallel architectures is challenging. These discrepancies worsen on heterogeneous architectures – such as clusters composed of standard CPUs in conjunction with GPU accelerators and/or Intel Xeon Phi co-processors –, which combine together different programming environments that may obey various floating-point models and offer different intermediate precision or different operators. Such non-determinism and non-reproducibility of floating-point computations on parallel machines causes validation and debugging issues, and may even lead to deadlocks.
Existing solutions to enhance reproducibility of BLAS routines are

- *Fixed reduction scheme* such as Intel's "Conditional Numerical Reproducibility" (CNR) available as an option in the Math Kernel Library (MKL);
- Avoid rounding error by using the *Kulisch accumulator* [3];
- *Mixed solution* as the one proposed by Demmel and Nguyen for BLAS level-1.

Our Approach

We introduced in [1] an approach to compute deterministic sums of floating-point numbers. This approach is based on a *multi-level algorithm* that combines efficiently floating-point expansion [2], which are placed in registers, and Kulisch accumulator.

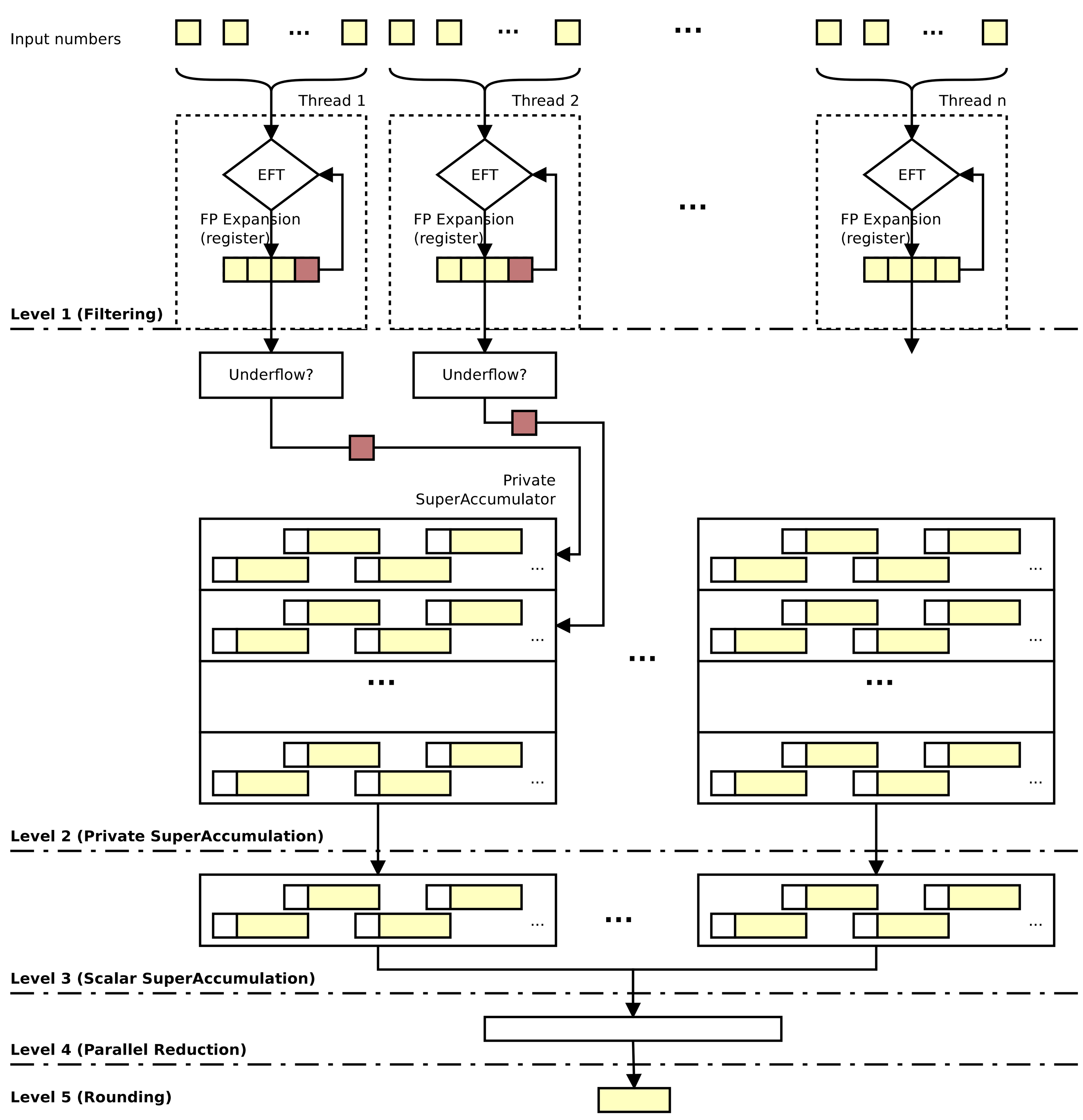

Fig. 1: Hierarchical superaccumulation scheme.

Performance Results

We provided implementations of the multi-level summation scheme on a range of parallel platforms: desktop and server CPUs, the Intel Xeon Phi many-core accelerator, and both NVIDIA and AMD GPUs. We relied on the parallel summation algorithm as well as exact multiplication to develop the fast, accurate, and reproducible implementations of fundamental linear algebra operations such as dot product, triangular solver, and matrix-matrix multiplication. We verified the accuracy of our implementations by comparing the computed results with the ones produced by the multiple precision MPFR library.

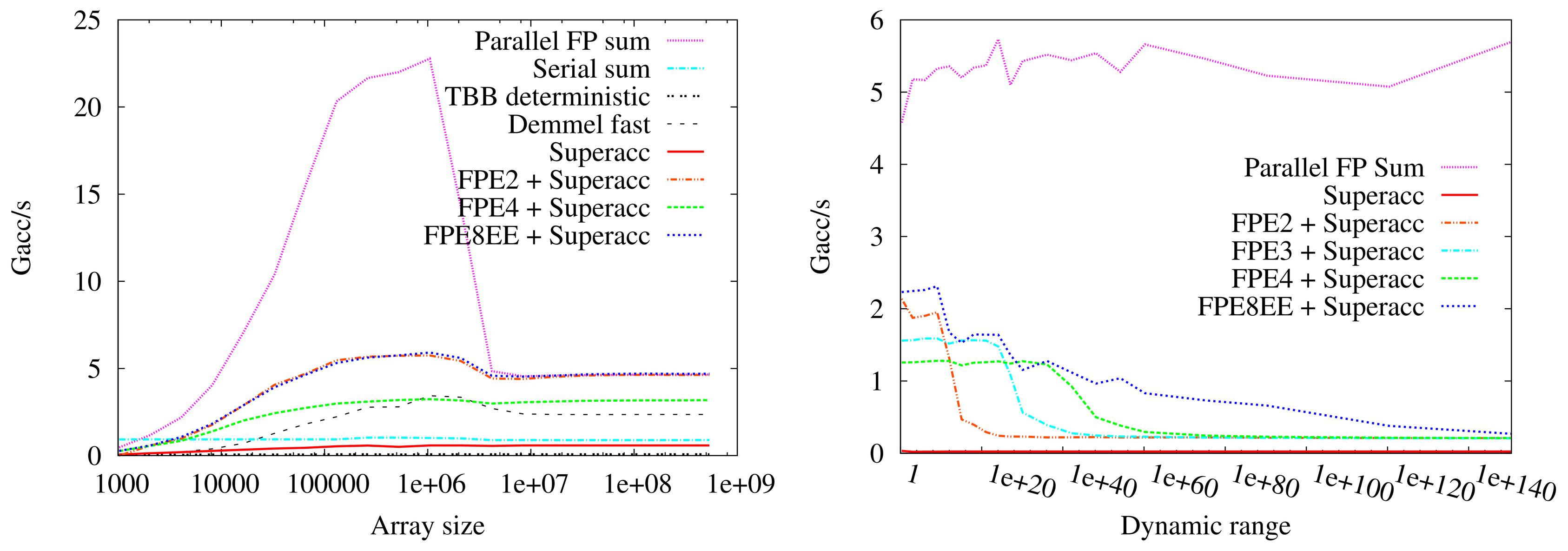

Fig. 2: The summation performance results on an 8-core Intel Xeon E5-4650L.

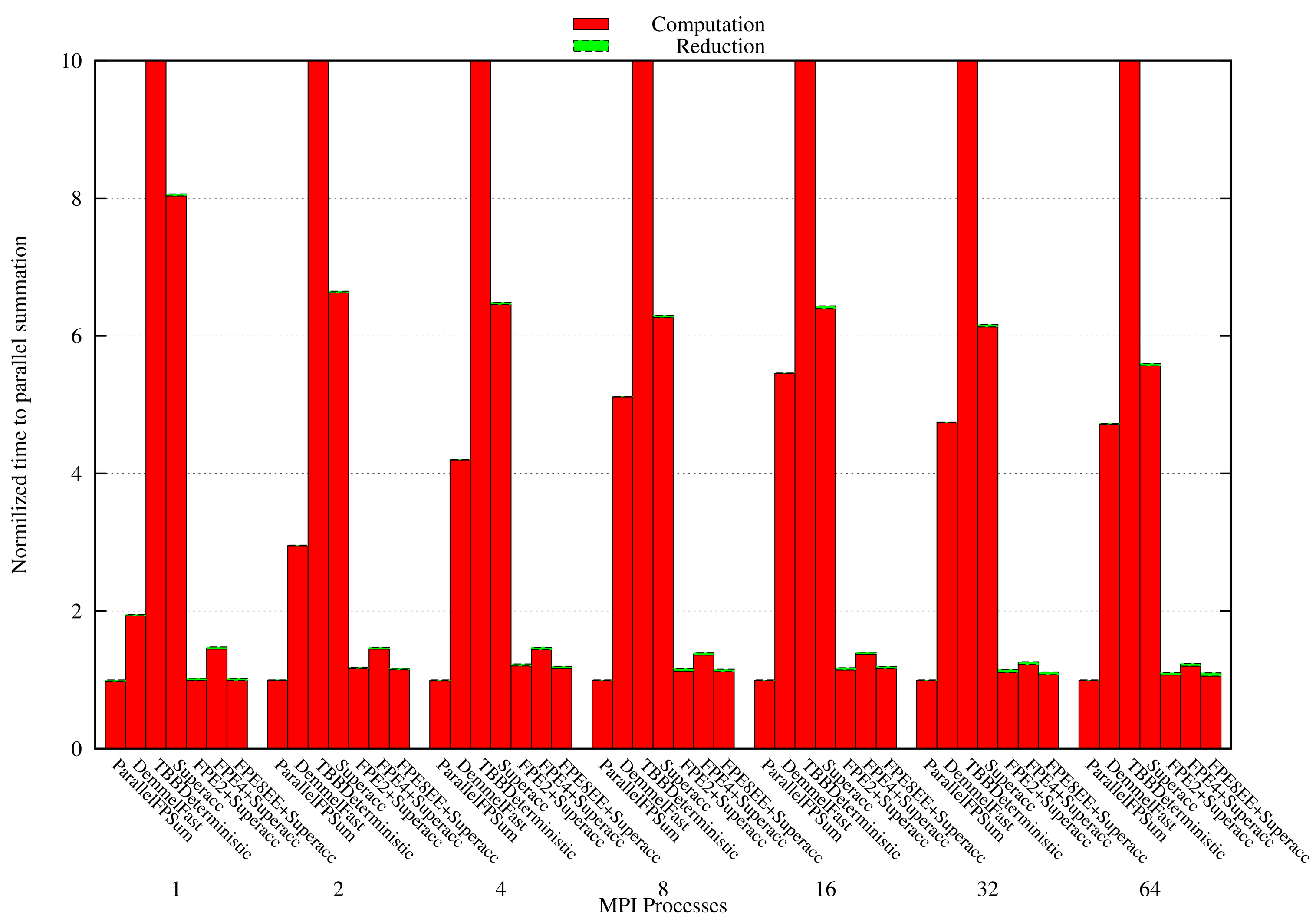

Fig. 3: Performance scaling on the Mesu cluster composed of 64 Intel Xeon E5-4650L nodes.

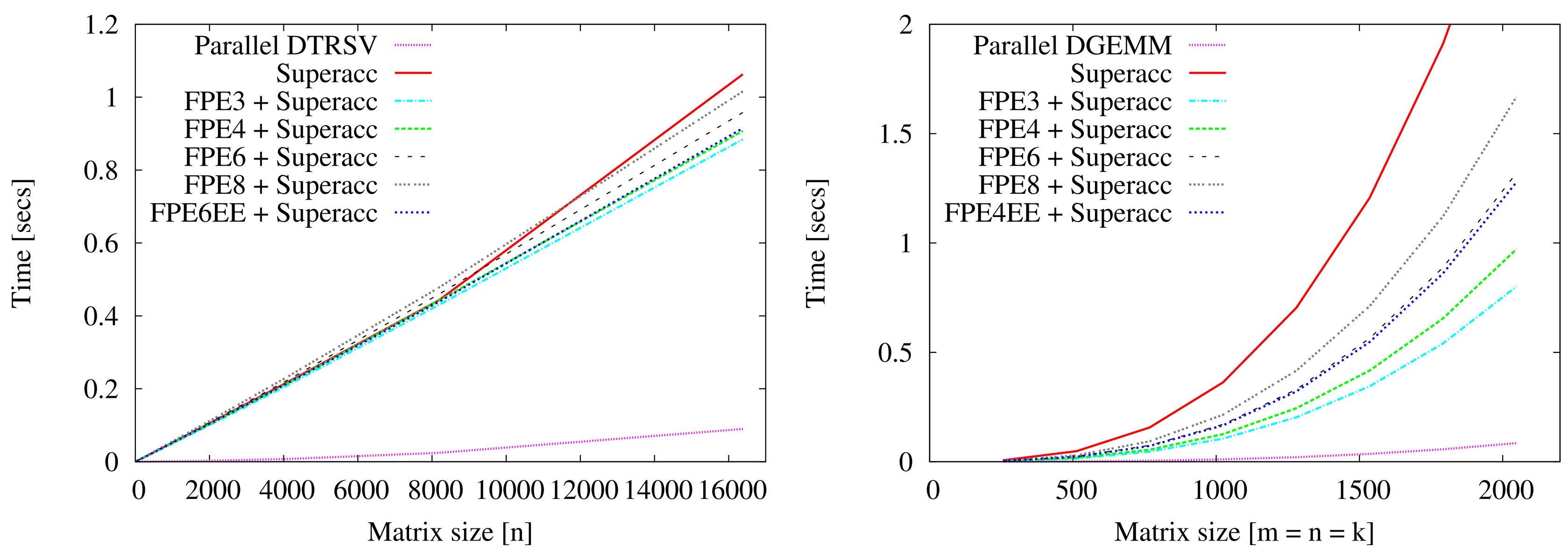

Fig. 4: The DTRSV and DGEMM performance results on NVIDIA K5000 and Tesla K20c, respectively.

Conclusions and Future Work

We presented an approach to achieve correct rounding for the floating-point summation problem, along with implementations on multi- and many-core architectures. This yielded results that are both reproducible and accurate to the last bit at no performance lost. We applied this approach to the other BLAS routines. Even though the performance of triangular solver and matrix product can be argued, their output is consistently reproducible and accurate independently of threads scheduling and data partitioning.
We plan to extend this approach to the rest of BLAS routines, derive specific implementations for compute-bound algorithms to be within 10 times performance overhead at most, and perform theoretical analysis of their accuracy.

Acknowledgement

This work undertaken (partially) in the framework of CALSIMLAB is supported by the public grant ANR-11-LABX-0037-01 overseen by the French National Research Agency (ANR) as part of the "Investissements d'Avenir" program (reference: ANR-11-IDEX-0004-02). This work was also granted access to the HPC resources of ICS financed by Region Île-de-France and the project Equip@Meso (reference ANR-10-EQPX-29-01).

References

[1] Sylvain Collange, David Defour, Stef Graillat, and Roman Iakymchuk. Full-Speed Deterministic Bit-Accurate Parallel Floating-Point Summation on Multi- and Many-Core Architectures. Technical Report HAL: hal-00949355, INRIA, DALI–LIRMM, LIP6, ICS, February 2014.

[2] Y. Hida, X. S. Li, and D. H. Bailey. Algorithms for quad-double precision floating point arithmetic. In *Proceedings of the 15th IEEE Symposium on Computer Arithmetic*, pages 155–162. IEEE Computer Society Press, Los Alamitos, CA, USA, 2001.

[3] U. W. Kulisch. *Computer arithmetic and validity*, volume 33 of *de Gruyter Studies in Mathematics*. Walter de Gruyter & Co., Berlin, 2nd edition, 2013. Theory, implementation, and applications.

web `https://exblas.lip6.fr/index.php` e-mail `roman.iakymchuk@lip6.fr`