High-Performance Computing Research at Leeds

with a focus on Computer Arithmetic

Massimiliano Fasi and Mantas Mikaitis

School of Computer Science, University of Leeds, Leeds, UK

School of Computer Science Research Day

January 16, 2025

Slides available: `https://mmikaitis.github.io`

What do we mean by *computer arithmetic*?

1. Pick a subset of real numbers (discretize part of the real number line), $\mathbb{F} \subset \mathbb{R}$.
2. Provide a mapping from $\mathbb{F}$ to a set of strings of bits.
3. Provide hardware or software to approximate basic operations: $+, -, \div, \surd, \times, \ldots$ (inputs/outputs strings of bits)
4. Provide hardware or software to approximate mathematical functions: exp, log, sin, cos, and others.
5. Provide hardware or software to approximate inner product and matrix multiplication.
6. Provide rounding modes.
7. Determine accuracy of operations and algorithms:
    - testing (no guarantees unless exhaustive), or
    - error analysis (bounds developed on paper).

IEEE 754: Most important standard in computing history

Standard for binary and decimal fixed-precision arithmetic

Defines subsets of reals, their encoding in memory, conversion and arithmetic behaviour, rounding, exception handling, and more. **Concept of correct rounding**.

Released in 1985, revised in

- 2008
- 2019 (active)
- 2029 (work in progress)

Leeds is participating in IEEE 754-2029

- Fortnightly meetings, discussion on the mailing list, thoroughly reading the 2019 revision and raising issues.
- MF acting as secretary: archiving minutes, organising activities.
- Working group: international, many members work in computing industry.

Floating-point arithmetic, main tools

A floating-point system $\mathbb{F} \subset \mathbb{R}$ is described with $\beta, t, e_{min}, e_{max}$ with elements

$$x = \pm m \times \beta^{e-t+1}.$$

Virtually all computers have $\beta = 2$ (binary FP).

Here t is precision, $e_{min} \leq e \leq e_{max}$ an exponent, $m \leq \beta^t - 1$ a significand $(m,\ t,\ e \in \mathbb{Z})$.

Standard model [Higham, 2002]

Given $x, y \in \mathbb{R}$ that lie in the range of $\mathbb{F}$ it can be shown that

$$\mathrm{fl}(x \mathbin{\mathrm{op}} y) = (x \mathbin{\mathrm{op}} y)(1+\delta), \quad |\delta| \leq u,$$

where $u = 2^{-t}$, $\mathrm{op} \in \{+, -, \times, \div\}$ and **round-to-nearest** mode.

Building error bounds: small example

Rounding errors δ accumulate. For example, consider computing $s = x_1 y_1 + x_2 y_2 + x_3 y_3$.

We compute $\widehat{s}$ with

$$\begin{aligned}\widehat{s} &= \Big(\big(x_1 y_1 (1+\delta_1) + x_2 y_2 (1+\delta_2)\big)(1+\delta_3) + x_3 y_3 (1+\delta_4)\Big)(1+\delta_5) \\ &= x_1 y_1 (1+\delta_1)(1+\delta_3)(1+\delta_5) + x_2 y_2 (1+\delta_2)(1+\delta_3)(1+\delta_5) + x_3 y_3 (1+\delta_4)(1+\delta_5).\end{aligned}$$

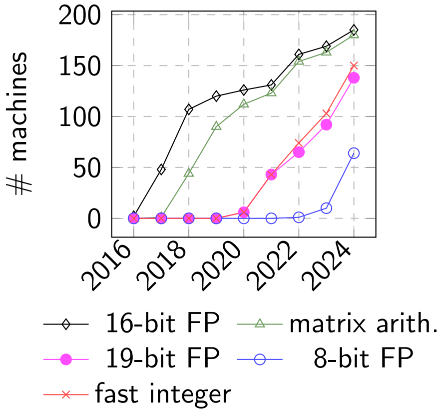
Therefore we compute a solution for the inputs *perturbed at most by* $\prod_{i=1}^{n}(1+\delta_i)$.

Worst case backward error bound

$\prod_{i=1}^{n}(1+\delta_i) = 1 + \theta_n, \quad |\theta_n| \leq \gamma_n$, with $\gamma_n = \frac{nu}{1-nu}$ and assuming $nu < 1$.

To simplify, we say worst-case error growth is $\mathcal{O}(nu)$.

Using the TOP500 to anticipate where HPC hardware is going



Devices counted: P100, V100, A100, H100, MI210, MI250X, MI300X, Intel Data Center GPU, from `https://www.top500.org`. With NVIDIA Blackwell 4/6-bit FP will appear.

Research direction 1: Using low precision floating-point matrix arith.

Architecture	Input format	Accumulation format
NVIDIA PTX ISA 8.5	fp8-E5M2	binary32
	fp8-E4M3	binary32
	binary16	binary16
	binary16	binary32
	bfloat16	binary32
	19-bit FP	binary32
AMD MI300 ISA	fp8-E5M2	binary32
	fp8-E4M3	binary32
	binary16	binary32
	bfloat16	binary32
	19-bit FP	binary32

Research direction 1: Using low precision floating-point matrix arith.

Mary and Mikaitis [2024]; Fasi et al. [2023]; Higham et al. [2019].

Approach 1: Use 8-bit FP directly (scale to avoid overflow)

$$C = \Lambda^{-1}\Big(\mathrm{fl}(\Lambda A)\mathrm{fl}(BM)\Big)M^{-1}$$

Approach 2: Use multiple 8-bit FP directly

$$A^{(i)} = \mathrm{fl}\left(\left(\Lambda A - \sum_{k=0}^{i-1} u^k A^{(k)}\right)/u^i\right)$$

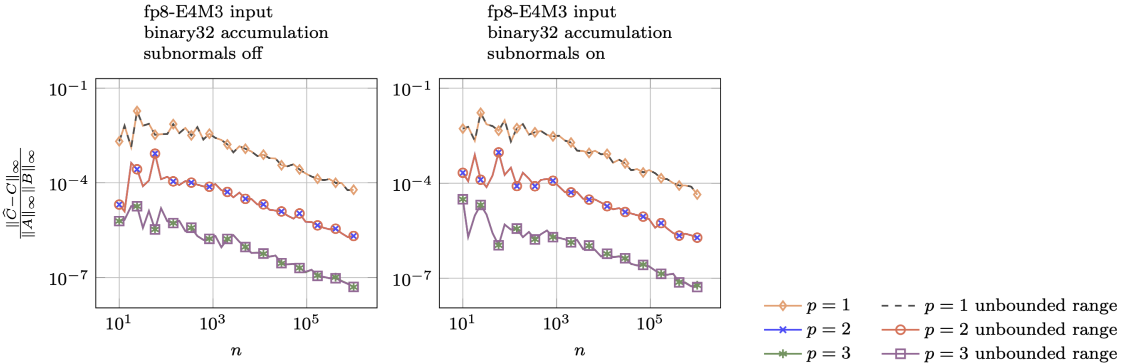
and we approximate $C = AB$ as

$$C \approx \Lambda^{-1}\left(\sum_{i+j<p} u^{i+j} A^{(i)} B^{(j)}\right) M^{-1}.$$

Research direction 1: Using low precision floating-point matrix arith.

Collaboration with T. Mary (Sorbonne Univ.) [2024].

Matrix multiply; data in $[-10^{10}, -10^{-10}] \cup [10^{-10}, 10^{10}]$.



Research direction 1: Using low precision integer matrix arith.

Collabor. with J. Dongarra, A. Abdelfattah (Univ. Tennessee), F. Tisseur (Manchester) [2025].

Ootomo et al. [2024] discovered algorithms for simulating FP matrix multiply with integer matrix multiply. Small example split:

$$\underset{A^T}{\begin{bmatrix} 2^0 \cdot 1.1001000 \\ 2^3 \cdot 1.0000000 \\ -2^1 \cdot 1.1101100 \end{bmatrix}} \Rightarrow 2^4 \cdot \underset{\text{Block fixed-point}}{\begin{bmatrix} \cancel{0}.\underline{000}\ \underline{110}\ \underline{010}\ \underline{000} \\ \cancel{0}.\underline{100}\ \underline{000}\ \underline{000}\ \underline{000} \\ -\cancel{0}.\underline{001}\ \underline{110}\ \underline{110}\ \underline{000} \end{bmatrix}} \Rightarrow 2^1 \cdot \underset{A^T_{(1)}}{\begin{bmatrix} 000 \\ 100 \\ -001 \end{bmatrix}} + 2^{-2} \cdot \underset{A^T_{(2)}}{\begin{bmatrix} 110 \\ 000 \\ -110 \end{bmatrix}} + 2^{-5} \cdot \underset{A^T_{(3)}}{\begin{bmatrix} 010 \\ 000 \\ -110 \end{bmatrix}} + 2^{-8} \cdot \underset{A^T_{(4)}}{\begin{bmatrix} 000 \\ 000 \\ 000 \end{bmatrix}}$$

$$\underset{B}{\begin{bmatrix} 2^0 \cdot 1.0110001 \\ -2^2 \cdot 1.1110100 \\ 2^1 \cdot 1.1101000 \end{bmatrix}} \Rightarrow 2^3 \cdot \underset{\text{Block fixed-point}}{\begin{bmatrix} \cancel{0}.\underline{001}\ \underline{011}\ \underline{000}\ \underline{100} \\ -\cancel{0}.\underline{111}\ \underline{101}\ \underline{000}\ \underline{000} \\ \cancel{0}.\underline{011}\ \underline{101}\ \underline{000}\ \underline{000} \end{bmatrix}} \Rightarrow 2^0 \cdot \underset{B^{(1)}}{\begin{bmatrix} 001 \\ -111 \\ 011 \end{bmatrix}} + 2^{-3} \cdot \underset{B^{(2)}}{\begin{bmatrix} 011 \\ -101 \\ 101 \end{bmatrix}} + 2^{-6} \cdot \underset{B^{(3)}}{\begin{bmatrix} 000 \\ 000 \\ 000 \end{bmatrix}} + 2^{-9} \cdot \underset{B^{(4)}}{\begin{bmatrix} 100 \\ 000 \\ 000 \end{bmatrix}}$$

Research direction 1: Using low precision integer matrix arith.

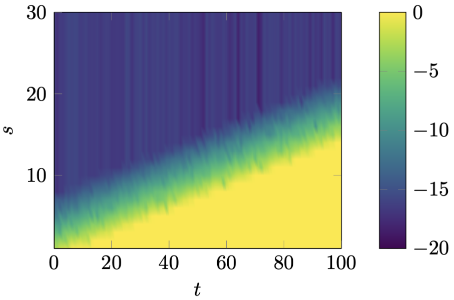
Collabor. with J. Dongarra, A. Abdelfattah (Univ. Tennessee), F. Tisseur (Manchester) [2025].

We are developing theoretical and experimental analysis of the algorithms. Analysis is general, to cover any future hardware changes.

Here s is the number of splits into 8-bit chunks; colour denotes forward error of (pow 10) $\frac{|a^T b - c|}{|c|}$, where

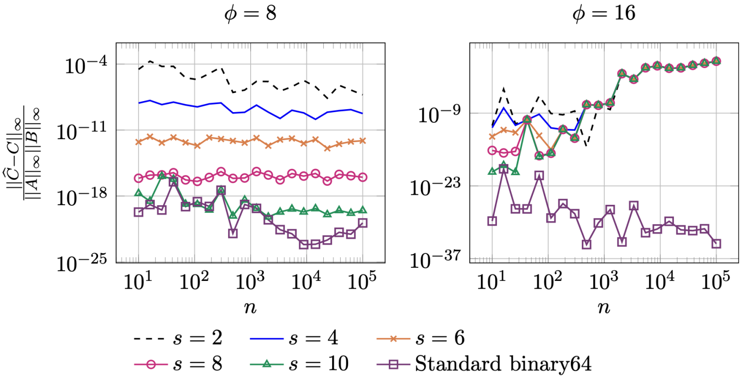
$$a = \begin{bmatrix} 2^{-t}x \\ 1 \end{bmatrix}, \qquad b = \begin{bmatrix} 2^{t}y \\ 1 \end{bmatrix},$$

and x, y are drawn from a uniform distribution and c is a reference solution.



Research direction 1: Using low precision integer matrix arith.

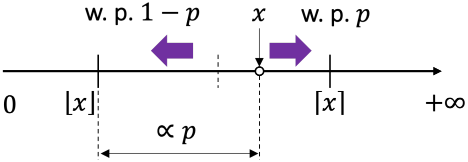
$$A_{ij}, B_{ij} = \mathrm{uniform}(-0.5, 0.5) \times e^{\phi\times \mathrm{normal}(0,1)}.$$



Research direction 2: Stochastic rounding

With **stochastic rounding** (**SR**), we are not rounding a number to the same direction, but to either direction with probability.

Given some x and FP neighbours $\lfloor x \rfloor$, $\lceil x \rceil$, we round to $\lceil x \rceil$ with prob. p and $\lfloor x \rfloor$ with $p - 1$.



Mode 1 SR (nearness): $p = \frac{x - \lfloor x \rfloor}{\lceil x \rceil - \lfloor x \rfloor}$ **Mode 2 SR**: $p = 0.5$

Mode 1

With **Mode 1 SR** we round x depending on its distances to the nearest two FP numbers, **cancelling out errors of different signs**.

Research direction 2: Rounding error analysis with SR

Standard error model for SR

With SR we replace u by $2u$ since it can round to the second nearest neighbour in $\mathbb{F}$.

Rounding error analysis

Worst-case error analysis determines the **upper bounds of errors**, while probabilistic error analysis describes **more realistic bounds**.

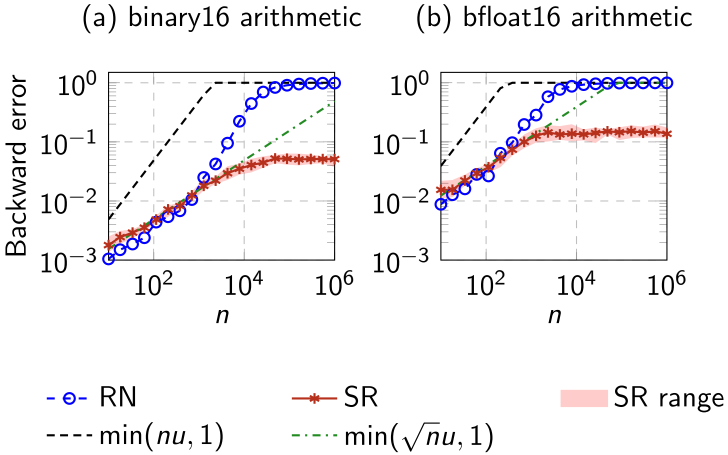
- Worst-case b-err bound with **RN**: $\frac{nu}{1-nu}$.
- Probabilistic bound with **RN**: $\lambda\sqrt{n}u + \mathcal{O}(u^2)$ w. p. $1 - 2ne^{-\lambda^2/2}$. Requires an assumption that δ_i are *mean independent zero-mean* quantities—do not always hold [Connolly, Higham, Mary, 2021].

Wilkinson rule of thumb

$\mathcal{O}(\sqrt{n}u)$ error growth is a rule of thumb with **RN**, but always holds with **SR**.

Research direction 2: Example error growth with SR in mat-vec prod

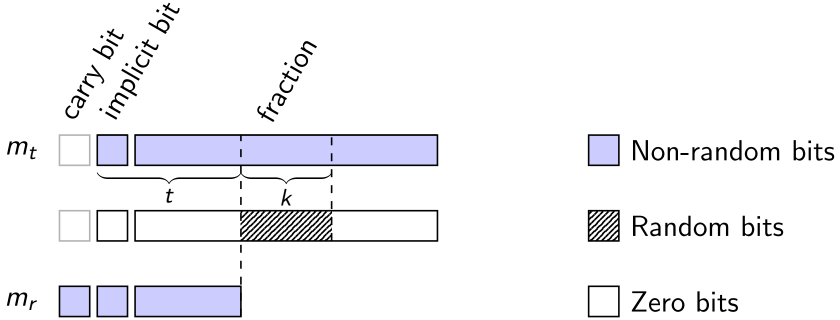
Backward error in $y = Ax$ where $A \in \mathbb{R}^{100\times n}$ with entries from uniform dist over $[0, 10^{-3}]$ and $x \in \mathbb{R}^n$ over $[0, 1]$: $\max_i \frac{|\widehat{y}-y|_i}{(|A||x|)_i}$.



Research direction 2: Consider implementation of SR

Take m_t to be a high precision unrounded significand from an operation.

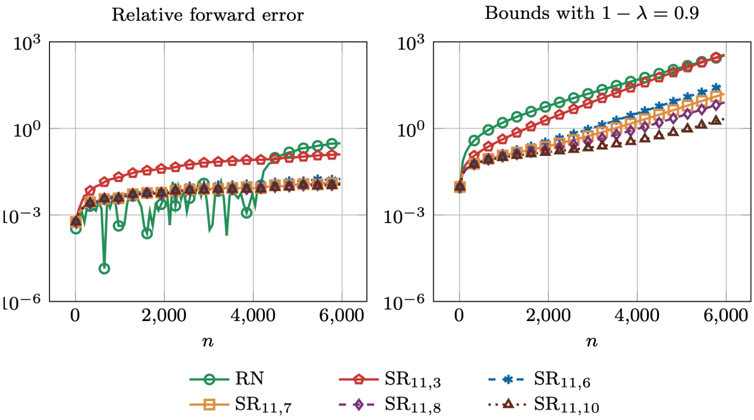
Take t to be source precision and k the precision of random numbers.



Research direction 2: Implementation of SR

E.-M. El Arar et al. [2024]: we derived a new bound $\mathcal{O}(\sqrt{n}u_p + nu_{p+r})$

Error when adding n random values in $(0, 1)$ in 16-bit floating point:



IEEE floating-point standardisation work: IEEE P3109

Standard for Arithmetic Formats for Machine Learning

New IEEE standard for computer arithmetic for AI is in progress. We are also actively participating in it: meeting fortnightly.

Standardises:

- Small floating-point subsets of reals (no -0, one NaN),
- encoding in 8-bit words,
- rounding behaviour,
- arithmetic operations needed in AI workloads,
- conversion to/from 754,
- exception handling.

Interim report available: `http://bit.ly/42gPWcy`

Research direction 3: Determining numerical features by testing hardware

Given a small matrix multiplier not necessarily using IEEE 754 $+, \times$:

$$\begin{array}{ccccccc} D & = & C & + & A & \times & B, \\ \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}}_{\substack{\text{binary16 or} \\ \text{binary32}}} & = & \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}}_{\substack{\text{binary16 or} \\ \text{binary32}}} & + & \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}}_{\text{binary16}} & \times & \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}}_{\text{binary16}} \end{array}$$

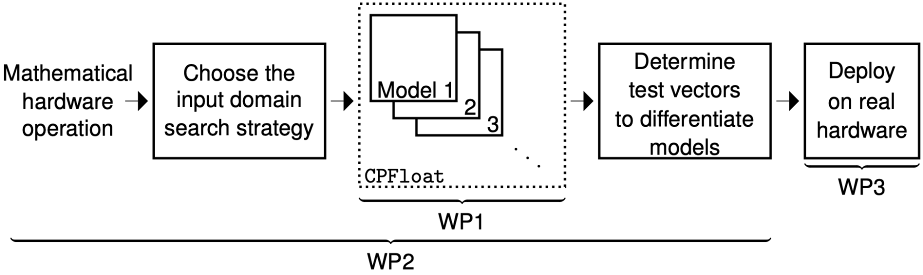
We are building theory and tools that allow to *report the internal precision, order of computations, rounding mode, and more*. Fasi, Higham, Pranesh, Mikaitis [2021].

We reported bit level differences between NVIDIA V100 and A100, not documented.

Research direction 3: Looking for numerical features by testing hardware

3-year EPSRC-funded project to start in 2025.

Collaboration with Argonne National Lab and Intel (US).



WP4: Develop automated open-source software to report hardware features.

Summary

- We are building theory and tools to explain computer arithmetic hardware and analyse algorithms.
- People have been there before: pre-IEEE-754 1985.
- More complicated now: vector and matrix hardware.
- Standardisation will impact future low-precision hardware.

Slides and more info at `http://mmikaitis.github.io`

References I

- N. J. Higham
  Accuracy and Stability of Numerical Algorithms. 2nd edition
  SIAM. 2002
- T. Mary and M. Mikaitis
  Error Analysis of Matrix Multiplication with Narrow Range Floating-Point Arithmetic
  hal-04671474. Aug. 2024
- M. Fasi, N. J. Higham, F. Lopez, T. Mary, and M. Mikaitis
  Matrix Multiplication in Multiword Arithmetic: Error Analysis and Application to GPU Tensor Cores
  SIAM J. Sci. Comput., 45:1. Feb. 2023.
- N. J. Higham, S. Pranesh, and M. Zounon
  Squeezing a Matrix into Half Precision, with an Application to Solving Linear Systems
  SIAM J. Sci. Comput., 41:4. Aug. 2019.

References II

A. Abdelfattah, J. Dongarra, M. Fasi, M. Mikaitis, and F. Tisseur
Error Analysis of Floating-Point Matrix Multiplication Simulated with Low-Precision Integer Arithmetic
In preparation. 2025.

H. Ootomo, K. Ozaki, and R. Yokota
DGEMM on integer matrix multiplication unit
Int. J. High Perform. Comput. Appl., 38:4. Mar. 2024.

M. P. Connolly, N. J. Higham, T. Mary
Stochastic rounding and its probabilistic backward error analysis.
SIAM J. Sci. Comput., 43. 2021.

E.-M. El Arar, M. Fasi, S.-I. Filip, and M. Mikaitis
Probabilistic Error Analysis of Limited-Precision Stochastic Rounding
hal-04665809. Jul. 2024.

References III

M. Fasi, N. J. Higham, S. Pranesh, and M. Mikaitis
Numerical Behavior of NVIDIA Tensor Cores
PeerJ Comput. Sci., 7. 2021

M. Fasi and M. Mikaitis
CPFloat: A C library for Simulating Low-Precision Arithmetic
ACM Trans. Math. Software, 49. 2023