The Limits of Matrix Computations at Extreme Scale and Low Precisions

Nick Higham
Department of Mathematics
The University of Manchester
https://nhigham.com

Slides available at https://bit.ly/hksiam22

Distinguished Lecture for HKSIAM and Hong Kong Universities

The Limits of What We Can Compute

$n \times n$ matrix prob.: rounding error bound $f(n)u$.

- ★ **Problem dimension** n
- ★ **Unit roundoff** u

both getting **larger**.

Increasingly **mixed precision world**:
$u < u_1 < u_2 \cdots$.

The Limits of What We Can Compute

$n \times n$ matrix prob.: rounding error bound $f(n)u$.

★ **Problem dimension** n

★ **Unit roundoff** u

both getting **larger**.

Increasingly **mixed precision world**:
$u < u_1 < u_2 \cdots$.

- **What can we guarantee about the computed solution?**

TOP500: November 2021

- **Fugaku** at Riken, Japan
- 158,976 A64FX Fujitsu/ARM v8.2-A CPUs
- Peak 537 petaflops.
- IEEE **double** $u \approx 1.1 \times 10^{-16}$, **half** $u \approx 4.9 \times 10^{-4}$.

	Rate	n
HPL	442 petaflops	2.1×10^7
HPL-AI	2.00 "**exaflops**"	1.6×10^7

Petaflops = 10^{15} *flops*
Exaflops = 10^{18} *flops*

Growth of Problem Size in TOP500

Dimension of matrix for #1 machine.

Machine	Date	n
Fugaku	2021	2.1×10^7
Jaguar	2010	6.3×10^6
ASCI RED	2000	3.6×10^5
CM-5/1024	1993	5.2×10^4

- Growing by roughly a **factor 10 every decade**.

Today's Floating-Point Arithmetics

		Bits			
Type	Name	Signif. (t)	Exp.	Range	$u = 2^{-t}$
Quarter	fp8-e5m2	3	5	$10^{\pm 5}$	1.2×10^{-1}
Quarter	fp8-e4m3	4	4	$10^{\pm 2}$	6.2×10^{-2}
Half	bfloat16	8	8	$10^{\pm 38}$	3.9×10^{-3}
Half	fp16	11	5	$10^{\pm 5}$	4.9×10^{-4}
Single	fp32	24	8	$10^{\pm 38}$	6.0×10^{-8}
Double	fp64	53	11	$10^{\pm 308}$	1.1×10^{-16}

- fp8 types introduced on NVIDIA H100 (2022).
- Bfloat16 used by Google TPU, Arm, Intel.
- Fp16 used by NVIDIA GPUs, AMD Radeon Instinct GPUs, ARM NEON, Fujitsu A64FX ARM.

Backward Error Analysis for LU Factorization

Let $\gamma_n = \dfrac{nu}{1 - nu} = nu + O(u^2)$.

Theorem

Computed solution $\widehat{x}$ to $Ax = b$ where $A \in \mathbb{R}^{n\times n}$ satisfies

$$(A + \Delta A)\widehat{x} = b, \quad |\Delta A| \le \gamma_{3n}|\widehat{L}||\widehat{U}|.$$

Then for $n \approx 10^7$:

- in IEEE double precision, $nu \approx 2.3 \times 10^{-9}$.
- in IEEE single precision, $nu \approx 1.25$.

Sharper Bound

Proof uses $A + \Delta A_1 = \widehat{L}\widehat{U}$, where (recall $\gamma_n \approx nu$),

$$|\Delta A_1| \le \begin{bmatrix} \gamma_1 & \gamma_1 & \cdots & \cdots & \gamma_1 \\ \gamma_1 & \gamma_2 & \cdots & \cdots & \gamma_2 \\ \vdots & \vdots & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \gamma_{n-1} & \gamma_{n-1} \\ \gamma_1 & \gamma_2 & \cdots & \gamma_{n-1} & \gamma_n \end{bmatrix} \circ |\widehat{L}||\widehat{U}|. \qquad (*)$$

Not fruitful to try to use $(*)$.

Low Precision in Deep Learning

- “We find that very low precision is sufficient not just for running trained networks but also for training them.” —**Courbariaux, Benji & David** (2015)

- “Deep learning models . . . are very tolerant of reduced-precision computations.”—**Dean (2019)**.

$$|\,\mathrm{fl}(x^T y) - x^T y| \le nu|x|^T|y|.$$

fp16: $nu = 1$ for $n = 2048$

bfloat16: $nu = 1$ for $n = 256$

- Yet deep learning successfully uses half precision.

The (Partial) Explanation

- **Inner products** not computed in the obvious way but are **blocked** $\Rightarrow$ much smaller error bounds possible.
- We use blocked algorithms.
- Hardware features automatically boost accuracy.
- The rounding error bounds are **worst-case** and **very pessimistic**. **Probabilistic error bounds** are more insightful.

The (Partial) Explanation

- **Inner products** not computed in the obvious way but are **blocked** $\Rightarrow$ much smaller error bounds possible.
- We use blocked algorithms.
- Hardware features automatically boost accuracy.
- The rounding error bounds are **worst-case** and **very pessimistic**. **Probabilistic error bounds** are more insightful.

Blocking is done for speed but also improves accuracy.

Blocked Inner Products: 2 Pieces

Original

$$s = \sum_{i=1}^{n} x_i y_i \;\Rightarrow\; |s - \widehat{s}| \le nu|x|^T|y|.$$

Blocked Inner Products: 2 Pieces

Original

$$s = \sum_{i=1}^{n} x_i y_i \;\Rightarrow\; |s - \widehat{s}| \le nu|x|^T|y|.$$

Blocked, 2 pieces

Let $n = 2b$.

$$s_1 = x(1\colon b)^T y(1\colon b)$$
$$s_2 = x(b+1\colon n)^T y(b+1\colon n)$$
$$s = s_1 + s_2$$

$$|s - \widehat{s}| \le \left(\frac{n}{2} + 1\right) u|x|^T|y|.$$

Blocked Inner Products; k Pieces

Original

$$s = \sum_{i=1}^{n} x_i y_i \;\Rightarrow\; |s - \widehat{s}| \le nu|x|^T|y|.$$

Blocked, k pieces

Let $n = kb$.

$$s_i = x((i-1)b+1\colon ib)^T y((i-1)b+1\colon ib),\; i = 1\colon k$$

$$s = s_1 + s_2 + \cdots + s_k$$

$$|s - \widehat{s}| \le \left(\frac{n}{k} + k - 1\right) u|x|^T|y|.$$

Blocked Inner Products; k Pieces

Original

$$s = \sum_{i=1}^{n} x_i y_i \;\Rightarrow\; |s - \widehat{s}| \le nu|x|^T|y|.$$

Blocked, k pieces

Let $n = kb$.

$$s_i = x((i-1)b + 1\colon ib)^T y((i-1)b + 1\colon ib),\ i = 1\colon k$$
$$s = s_1 + s_2 + \cdots + s_k$$

$$|s - \widehat{s}| \le \left(\frac{n}{k} + k - 1\right) u|x|^T|y|.$$

Optimal $k = \sqrt{n}$:

$$|s - \widehat{s}| \le 2\sqrt{n}\,u|x|^T|y|.$$

Block Summation

Recursive summation of $x_1, \dots, x_n$:

1 $s = 0$
2 for $i = 1\colon n$, $s = s + x_i$, end

Standard block summation:

1. sum blocks of size b by recursive summation: $(b-1)n/b = n - n/b$ additions
2. sum n/b partial sums by recursive summation: $n/b - 1$ additions

Idea: use a **more accurate method** in step 2.
E.g., recursive summation at *higher precision*, *compensated summation*.

FABsum

Blanchard, H & Mary (2020).

Input: n-vector x, block size b,
algs **FastSum**, **AccurateSum**.

1: **for** $i = 1: n/b$ **do**
2: Compute $s_i = \sum_{j=(i-1)b+1}^{ib} x_j$ with **FastSum**.
3: **end for**
4: Compute $s = \sum_{i=1}^{n/b} s_i$ with **AccurateSum**.

- **FastSum** is doing $n - n/b$ additions.
- **AccurateSum** is doing $n/b - 1$ additions.

FABsum Error Bound

$$\textbf{FastSum}: \quad \widehat{s} = \sum_{i=1}^{n} x_i(1+\mu_i^f), \quad |\mu_i^f| \le \epsilon_f(n),$$

$$\textbf{AccurateSum}: \quad \widehat{s} = \sum_{i=1}^{n} x_i(1+\mu_i^a), \quad |\mu_i^a| \le \epsilon_a(n).$$

Theorem

*The computed $\widehat{s}$ from **FABsum** satisfies*

$$\widehat{s} = \sum_{i=1}^{n} x_i(1+\mu_i),$$

$$|\mu_i| \le \epsilon(n,b) = \epsilon_f(b) + \epsilon_a(n/b) + \epsilon_f(b)\epsilon_a(n/b).$$

Error Bound for Recursive/Compensated

Take **FastSum** = recursive summation,
AccurateSum = compensated summation. Then

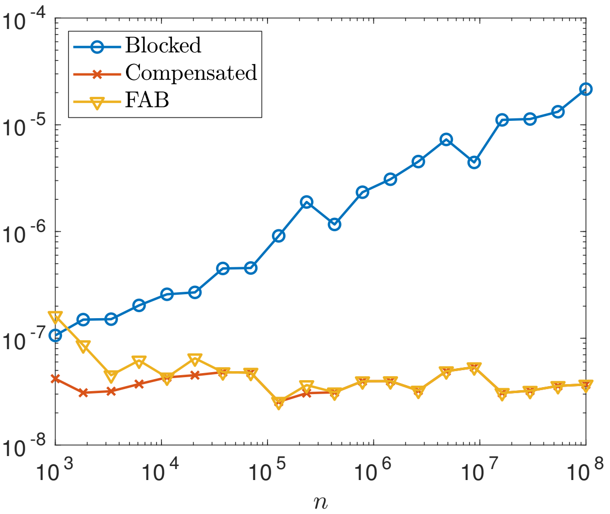
$$\epsilon(n,b) = (b+1)u + \left[4n/b + 2 + (b-1)^2 + 2(b-1)\right] u^2 + O(u^3).$$

Recall error bound is

- $nu + O(u^2)$ for recursive summation,
- $(n/b)u + O(u^2)$ for blocked summation.

FABsum error bound **independent of** n to first order.

Random Uniform [0, 1], $b = 128$, fp32



Blocked Matrix Multiplication

Let $A, B \in \mathbb{R}^{n \times n}$ be partitioned into $b \times b$ blocks A_{ij} and B_{ij}, where $p = n/b$ is assumed to be an integer. This algorithm computes $C = AB$.

```
for i = 1: p
    for j = 1: p
        C_ij = 0
        for k = 1: p
            X = A_ik B_kj
            C_ij = C_ij + X
        end
    end
end
```

- Compare $c_{rs} \leftarrow c_{rs} + a_{rk} b_{ks}$.

Blocked Algorithms

LAPACK philosophy: blocked matrix factorizations with a block size $b = 128$ or $b = 256$.
$\Rightarrow$ **Reduction in error bounds by factor b.**
At block level, apply block inner products giving further reduction!

- LAPACK manual states error bounds $p(n)u$, where "*$p(n)$ is a modestly growing function of n*".
- Standard NLA refs don't mention b in error bounds.
  - Optimizing constants not the point (Wilkinson).
  - Constants depend on the block alg.
  - Analysis is more complicated.

Extended Precision Registers

- **Intel x86-64** processors include 80-bit floating point registers with 64-bit significand (but not used by SSE2).
- Registers have $u = 2^{-64}$ rather than $u = 2^{-53}$ for double precision. Error bounds smaller by a factor up to $2^{11} = 2048$.
- **Caveat**: extra precision registers can lead to strange rounding effects, including double rounding!

Fused Multiply-Add (FMA)

Computes $x + yz$ at same speed as "$+$" or "$*$" with just one rounding error.

Without an FMA,

$$\mathrm{fl}(x + yz) = \big(x + yz(1 + \delta_1)\big)(1 + \delta_2), \quad |\delta_1|, |\delta_2| \le u,$$

but **with an FMA**

$$\mathrm{fl}(x + yz) = (x + yz)(1 + \delta), \quad |\delta| \le u.$$

Error bounds for inner product-based computations **reduced by a factor 2.**

Mixed Precision Block FMA

Precisions $u_{\rm low}$ (fp8, bfloat16, fp16), $u_{\rm high}$ (fp16, fp32).

Dimensions:

$$\underbrace{D}_{b_1\times b_2} = \underbrace{C}_{b_1\times b_2} + \underbrace{A}_{b_1\times b}\,\underbrace{B}_{b\times b_2}.$$

Precisions:

$$\underbrace{D}_{u_{\rm low} \text{ or } u_{\rm high}} = \underbrace{C}_{u_{\rm low} \text{ or } u_{\rm high}} + \underbrace{A}_{u_{\rm low}}\ \underbrace{B}_{u_{\rm low}}.$$

Computation:

$$\mathrm{fl}_{\mathrm{high}}\Big(C + \mathrm{fl}_{\mathrm{high}}(AB)\Big).$$

Can chain: $C \leftarrow C + AB$.

Block FMA Hardware

Year	Device	Dimensions	u_{low}	u_{high}
2020	Google TPU v4i	$128 \times 128 \times 128$	bfloat16	fp32
2017	NVIDIA V100	$4 \times 4 \times 4$	fp16	fp32
2019	ARMv8.6-A	$2 \times 4 \times 2$	bfloat16	fp32
2020	NVIDIA A100	$8 \times 8 \times 4$	bfloat16	fp32
		$8 \times 8 \times 4$	fp16	fp32
		$8 \times 4 \times 4$	TFloat-32	fp32
		$2 \times 4 \times 2$	fp64	fp64

Note

- Not necessarily IEEE compliant.
- Very fast throughput (*"one result per cycle"*) compared with none block-FMA arithmetic.

Error Analysis of Block FMAs

Blanchard, H , Lopez, Mary, & Pranesh (2020).

Analysis of algs for **matrix mult** $C = AB$ based on block FMA. *Inherently multiprecision.*

For $A, B \in \mathbb{R}^{n\times n}$ using chained block $b \times b$ FMAs,

$$|C - \widehat{C}| \le f(n, b, u_{\mathrm{low}}, u_{\mathrm{high}})|A||B|,$$

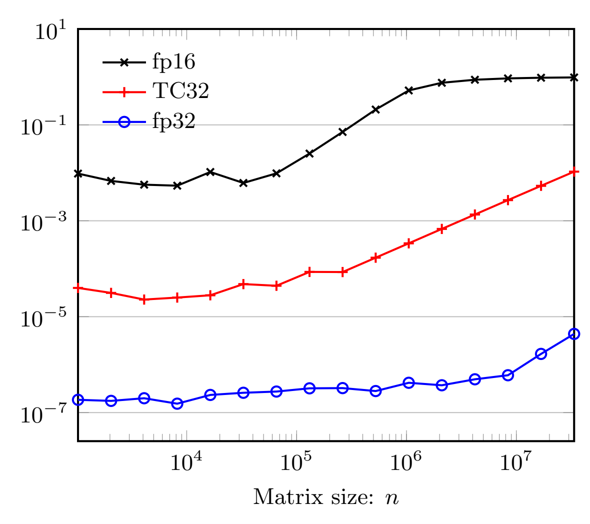
where with A, B given in u_{high}, $f(\cdot)$ is

Standard in precision u_{low}	$(n + 2)u_{\mathrm{low}}$
Block FMA, u_{high} internally	$2u_{\mathrm{low}} + nu_{\mathrm{high}}$
Standard in precision u_{high}	nu_{high}

- Similar results for **LU factorization** and $Ax = b$.

NVIDIA V100

- Matrix entries are rand unif $[0, 10^{-3}]$.
- In fp32, cmp'wise error $\max_{i,j} |\widehat{C} - C|_{ij}/(|A||B|)_{ij}$:



Probabilistic Error Analysis

Rounding error bounds above are **worst-case**.

> ***"To be realistic, we must prune away the unlikely. What is left is necessarily a probabilistic statement."***
>
> ***— Stewart, 1990***

Statistical Effects

"In general, the statistical distribution of the rounding errors will reduce considerably the function of n ***occurring in the relative errors. We might expect in each case that this function should be replaced by something which is no bigger than its square root."***

— Wilkinson, 1961

Statistical Effects

"In general, the statistical distribution of the rounding errors will reduce considerably the function of** n **occurring in the relative errors. We might expect in each case that this function should be replaced by something which is no bigger than its square root."

— Wilkinson, 1961

Limitations of central limit theorem argument

- Rounding errors **independent** random variables of **mean zero**.
- Applies only to **first-order** part of error.
- n is **sufficiently large**.

Standard Tool for Rounding Error Analysis

Theorem

If $|\delta_i| \le u$ and $\rho_i = \pm 1$ for $i = 1:n$ and $nu < 1$ then

$$\prod_{i=1}^{n} (1 + \delta_i)^{\rho_i} = 1 + \theta_n,$$

where

$$|\theta_n| \le \gamma_n := \frac{nu}{1 - nu} = nu + O(u^2).$$

- The basis of most rounding error analyses.
- We seek an analogous result with a smaller, but **probabilistic**, bound on θ_n.

Assumptions for Probabilistic Analysis

Model M

- **Rounding error bound**:

  $$\mathrm{fl}(x \,\mathrm{op}\, y) = (x \,\mathrm{op}\, y)(1+\delta), \quad |\delta| \le u, \quad \mathrm{op} \in \{+,-,*,/\}.$$

- **Mean independence**:
  The computation generates $\delta_1, \delta_2, \ldots$ that are random variables of mean zero such that

  $$\mathbb{E}(\delta_{k+1} \mid \delta_1, \ldots, \delta_k) = \mathbb{E}(\delta_{k+1}) = 0.$$

- Weaker than assuming the δ_i are independent.
- The δ_i need not be from same distribution.

Probabilistic Analysis

Theorem (Connolly, H & Mary, 2021)

Let $\delta_1, \ldots, \delta_n$ satisfy Model M. For any constant $\lambda > 0$ and $\rho_i = \pm 1$, $i = 1 : n$,

$$\prod_{i=1}^{n} (1 + \delta_i)^{\rho_i} = 1 + \theta_n, \quad |\theta_n| \le \widetilde{\gamma}_n(\lambda) \approx \lambda \sqrt{n} u,$$

holds with probability at least $1 - 2\exp(-\lambda^2/2)$.

- Proof uses martingales.
- Valid for all n.
- Valid to all orders.
- Explicit probability $P(\lambda)$ (pessimistic).
- Earlier result by **H & Mary (2020)** assumes indep.

Inner Products

Theorem

Let $s = x^T y$, where $x, y \in \mathbb{R}^n$. Under Model M, the computed $\widehat{s}$ satisfies

$$\widehat{s} = (x + \Delta x)^T y,$$
$$|\Delta x| \le \widetilde{\gamma}_n(\lambda)|x| \approx \lambda\sqrt{n}u|x|,$$

with probability at least $1 - 2n\exp(-\lambda^2/2)$.

Similar results by **H & Mary (2020)**, **Ipsen & Zhou (2020)**.

Linear Systems

Theorem

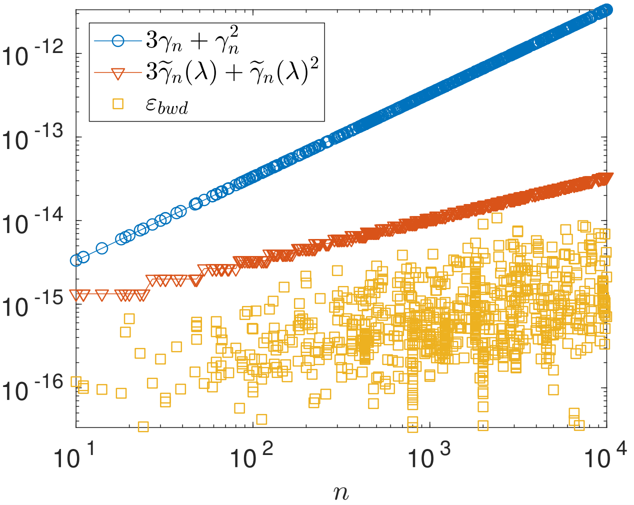
Under Model M, the computed solution $\widehat{x}$ to $Ax = b$ from LU factorization satisfies

$$(A + \Delta A)\widehat{x} = b, \quad |\Delta A| \le \bigl(3\widetilde{\gamma}_n(\lambda) + \widetilde{\gamma}_n(\lambda)^2\bigr)|\widehat{L}||\widehat{U}|,$$

with probability at least $1 - 2n^3/3\,\exp(-\lambda^2/2)$.

Real-Life Matrices

Solution of $Ax = b$ (fp64), b from Uniform $[0, 1]$,
for 943 matrices from **SuiteSparse** collection ($\lambda = 1$).



Probabilistic QR Error Bound

Theorem (Connolly & H, 2022)

Under Model M, for the computed $\widehat{R} \in \mathbb{R}^{m\times n}$ from Householder QR on $A \in \mathbb{R}^{m\times n}$ ($m \ge n$), $\exists$ orthogonal $Q \in \mathbb{R}^{m\times m}$ such that

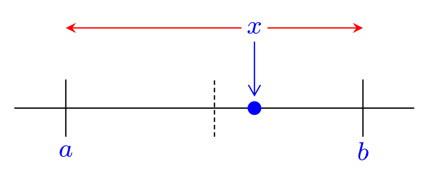
$$A + \Delta A = Q\widehat{R},$$
$$\|\Delta a_j\|_2 \le c\lambda\sqrt{mn}\,u\|a_j\|_2 + O(u^2), \quad j = 1\colon n,$$

holds with probability at least $1 - 2mn\exp(-\lambda^2)$.

- Worst-case bound has *mnu*.
- Square rooting of constant applies to Givens QR, too.

Stochastic Rounding

Forsythe (1950), ..., **Croci et al. (2022)**.

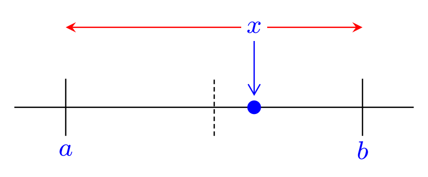


Theorem (Connolly, H & Mary, 2021)

The rounding errors $\delta_1, \delta_2, \dots$ from stochastic rounding are rand. vars of mean 0 s.t. $\mathbb{E}\big(\delta_k \mid \delta_1, \dots, \delta_{k-1}\big) = \mathbb{E}(\delta_k) = 0$.

Stochastic Rounding

Forsythe (1950), . . . , **Croci et al. (2022)**.



Theorem (Connolly, H & Mary, 2021)

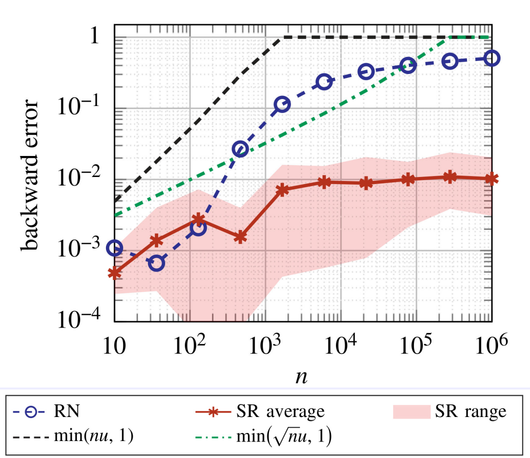
The rounding errors $\delta_1, \delta_2, \ldots$ *from stochastic rounding are rand. vars of mean* 0 *s.t.* $\mathbb{E}\big(\delta_k \mid \delta_1, \ldots, \delta_{k-1}\big) = \mathbb{E}(\delta_k) = 0$.

Stochastic rounding always satisfies the assumptions!

For SR, we can *always* replace nu by $\sqrt{n}u$ in a worst-case rounding error bound to obtain a probabilistic error bound.

Stagnation

Harmonic sum $\sum_{k=1}^{n} 1/k$ in fp16.



Stochastic rounding avoids *stagnation*!

Random Data

Model M'

- d_j, $j = 1\colon n$, are independent random variables from a distribution of mean μ_x s.t. $|d_j| \le \xi_d$, $j = 1\colon n$.
- $\mathbb{E}(\delta_k \mid \delta_1, \ldots, \delta_{k-1}, d_1, \ldots, d_n) = \mathbb{E}(\delta_k) = 0.$

Theorem (H & Mary, 2020)

If $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times p}$ Model M', with means μ_A, μ_B and bounds ξ_A, ξ_B, and let $C = AB$. Under Model M',

$$\max_{i,j} |(C - \widehat{C})_{ij}| \le \bigl(\lambda|\mu_A\mu_B| n^{3/2} + (\lambda^2 + 1)\xi_A\xi_B n\bigr) u + O(u^2)$$

with probability at least $P(\lambda) = 1 - 2mnp\exp(-\lambda^2/2)$.

Putting It All Together

- Block algs reduce error bound by factor b.
- For blocking at multiple levels, the reduction factors can accumulate.
- Extended precision registers and (block) FMAs give automatic accuracy boost.
- Block size $b = 256$ and 80-bit registers reduces error bound by factor $256 \times 2048 = 5.2 \times 10^5$.
- Prob error anal. says "$f(n)u \to \sqrt{f(n)}u$".
- Prob. error anal. applies to blocked algs. Error constant $(b + n/b - 1)u$ for a blocked inner product translates to $(\sqrt{b} + \sqrt{n/b})u$ in a prob. bound.

Conclusions

- Classical analyses **no longer guarantee the numerical stability** of classical algorithms for all n and u of interest,
- **Block algs** (designed for speed) & **hardware features** give significant accuracy boosts.
- New **probabilistic bounds** show "$f(n)u \to \sqrt{f(n)}u$". Even these bounds often very pessimistic.
- We often do better than we can currently explain.

Slides at **`https://bit.ly/hksiam22`**

References I

- Pierre Blanchard, Nicholas J. Higham, Florent Lopez, Theo Mary, and Srikara Pranesh.
  Mixed precision block fused multiply-add: Error analysis and application to GPU tensor cores.
  SIAM J. Sci. Comput., 42(3):C124–C141, 2020.

- Pierre Blanchard, Nicholas J. Higham, and Theo Mary.
  A class of fast and accurate summation algorithms.
  SIAM J. Sci. Comput., 42(3):A1541–A1557, 2020.

- Michael P. Connolly, Nicholas J. Higham, and Theo Mary.
  Stochastic rounding and its probabilistic backward error analysis.
  SIAM J. Sci. Comput., 43(1):A566–A585, 2021.

References II

- Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David.
  BinaryConnect: Training deep neural networks with binary weights during propagations.
  In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, Curran Associates, Inc., 2015, pages 3123–3131.

- Matteo Croci, Massimiliano Fasi, Nicholas J. Higham, Theo Mary, and Mantas Mikaitis.
  Stochastic rounding: Implementation, error analysis and applications.
  Roy. Soc. Open Sci., 9(3):1–25, 2022.

References III

Jeffrey Dean.
The deep learning revolution and its implications for computer architecture and chip design.
ArXiv:1911.05289v1, November 2019.

George E. Forsythe.
Reprint of a note on rounding-off errors.
SIAM Rev., 1(1):66–67, 1959.

James W. Hanlon.
New chips for machine intelligence.
https://jameswhanlon.com/new-chips-for-machine-intelligence.html, October 2019.
Accessed November 27, 2019.

References IV

Nicholas J. Higham.
Numerical stability of algorithms at extreme scale and low precisions.
MIMS EPrint 2021.14, Manchester Institute for Mathematical Sciences, The University of Manchester, UK, September 2021.
21 pp.
To appear in Proc. Int. Cong. Math.

Nicholas J. Higham and Theo Mary.
A new approach to probabilistic rounding error analysis.
SIAM J. Sci. Comput., 41(5):A2815–A2835, 2019.

References V

Nicholas J. Higham and Theo Mary.
Sharper probabilistic backward error analysis for basic linear algebra kernels with random data.
SIAM J. Sci. Comput., 42(5):A3427–A3446, 2020.

Ilse C. F. Ipsen and Hua Zhou.
Probabilistic error analysis for inner products.
SIAM J. Matrix Anal. Appl., 41(4):1726–1741, 2020.

G. W. Stewart.
Stochastic perturbation theory.
SIAM Rev., 32(4):579–610, 1990.

J. H. Wilkinson.
Error analysis of direct methods of matrix inversion.
J. ACM, 8:281–330, 1961.