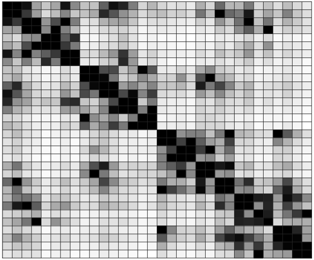
**Communications in NLA**
September 14, 2020

# Mixed Precision Low Rank Compression of Data Sparse Matrices

**Theo Mary**
Sorbonne Université, CNRS, LIP6
https://www-pequan.lip6.fr/~tmary/

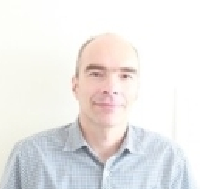
Slides available at https://bit.ly/CommNLA



# Collaborators

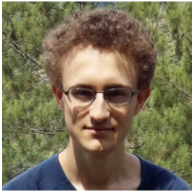
Patrick Amestoy

Olivier Boiteau



Alfredo Buttari

Matthieu Gerest



Fabienne Jézéquel

Jean-Yves L'Excellent

# Widening range of arithmetics

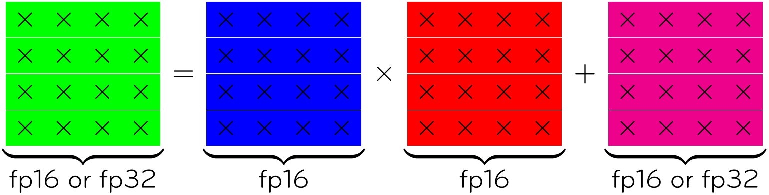
| | Bits Signif. ($t$) | Exp. | Range | $u = 2^{-t}$ |
|---|---|---|---|---|
| bfloat16 | 8 | 8 | $10^{\pm 38}$ | $4 \times 10^{-3}$ |
| fp16 | 11 | 5 | $10^{\pm 5}$ | $5 \times 10^{-4}$ |
| fp32 | 24 | 8 | $10^{\pm 38}$ | $6 \times 10^{-8}$ |
| fp64 | 53 | 11 | $10^{\pm 308}$ | $1 \times 10^{-16}$ |
| fp128 | 113 | 15 | $10^{\pm 4932}$ | $1 \times 10^{-34}$ |

Half precision increasingly supported by hardware:

- Fp16 used by NVIDIA GPUs, AMD Radeon Instinct MI25 GPU, ARM NEON, Fujitsu A64FX ARM
- Bfloat16 used by Google TPU, NVIDIA GPUs, Arm, Intel

# Benefits from low precisions

- Reduced storage and communications
- Increased speed, e.g., with GPU Tensor Cores

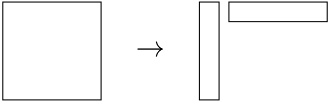



fp32 $\rightarrow$ fp16 speedup evolution:
P100: 2$\times$ V100: 8$\times$ A100: 16$\times$ (announced)

- Correspondingly low accuracy $\Rightarrow$ mixed precision algorithms
- Mixed precision algs. highly successful in NLA: linear systems, matrix factorizations, matrix multiplication, iterative methods, least squares, EVD, SVD, matrix functions, FFT, and many others (see some references at the end of the slides)

# Low rank compression

$$\underset{n\times n}{A} \approx \underset{n\times r}{X} \; \underset{r\times n}{Y^T}$$



- $\varepsilon$-rank of $A$:

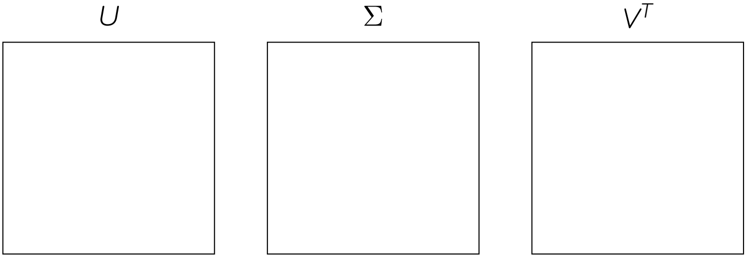
  smallest $r_\varepsilon$ such that $\exists T, \quad \mathrm{rank}(T) = r_\varepsilon, \quad \|A - T\| \le \varepsilon \|A\|$

- Optimal $\varepsilon$-approximation given by truncated SVD (Eckart–Young)

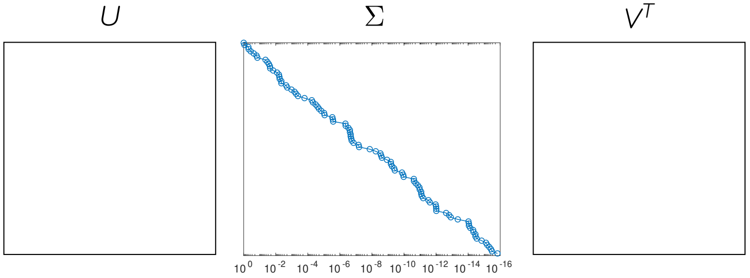
$$A = U\Sigma V^T \quad \Rightarrow \quad T = U_\varepsilon \Sigma_\varepsilon V_\varepsilon^T = \sum_{i=1}^{r_\varepsilon} u_i \sigma_i v_i^T$$

- **What precision should we store $T$ in ?**
- Naive answer: lowest possible precision with unit roundoff safely smaller than $\varepsilon$ (e.g., fp64 if $\varepsilon < u_{\mathrm{fp32}} \approx 6 \times 10^{-8}$)

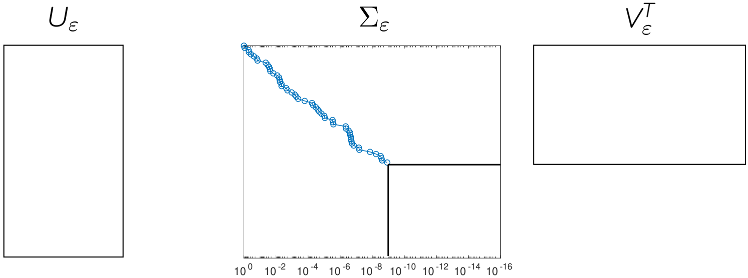
# Mixed precision SVD: an example



# Mixed precision SVD: an example

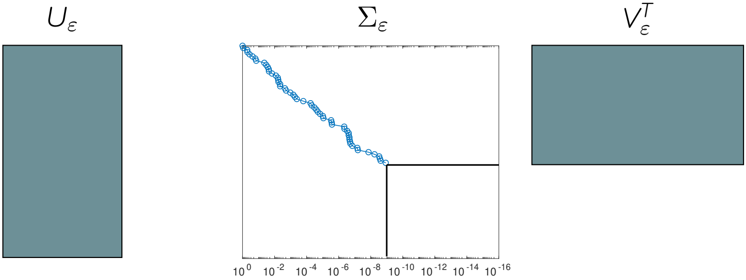


# Mixed precision SVD: an example



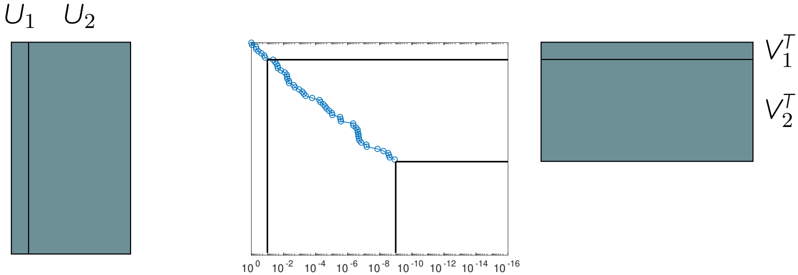

- Assume $\varepsilon = 10^{-9} \Rightarrow \|A - U_\varepsilon \Sigma_\varepsilon V_\varepsilon^T\| \le \varepsilon \|A\|$

# Mixed precision SVD: an example



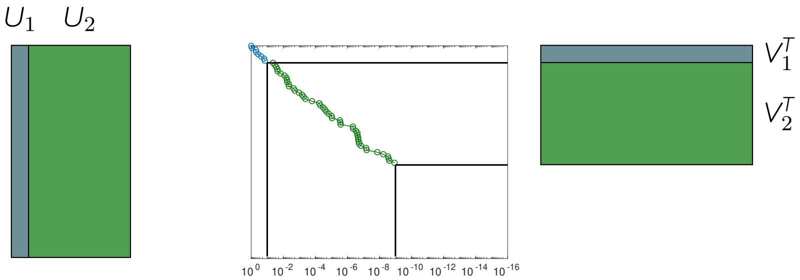

- Assume $\varepsilon = 10^{-9} \Rightarrow \|A - U_\varepsilon \Sigma_\varepsilon V_\varepsilon^T\| \leq \varepsilon \|A\|$
- Naive approach: use **double precision** because $u_{\text{fp32}} > \varepsilon$

# Mixed precision SVD: an example




- Assume $\varepsilon = 10^{-9} \Rightarrow \|A - U_\varepsilon \Sigma_\varepsilon V_\varepsilon^T\| \le \varepsilon \|A\|$
- Naive approach: use **double precision** because $u_{\mathrm{fp32}} > \varepsilon$
- Let $U_\varepsilon = [U_1 \; U_2]$, $\Sigma_\varepsilon = \mathrm{diag}(\Sigma_1, \Sigma_2)$, and $V_\varepsilon = [V_1 \; V_2]$, such that $\|\Sigma_2\| \le \varepsilon / u_{\mathrm{fp32}} \approx 2 \times 10^{-2}$

# Mixed precision SVD: an example




- Assume $\varepsilon = 10^{-9} \Rightarrow \|A - U_\varepsilon \Sigma_\varepsilon V_\varepsilon^T\| \leq \varepsilon \|A\|$
- Naive approach: use **double precision** because $u_{\text{fp32}} > \varepsilon$
- Let $U_\varepsilon = [U_1 \; U_2]$, $\Sigma_\varepsilon = \text{diag}(\Sigma_1, \Sigma_2)$, and $V_\varepsilon = [V_1 \; V_2]$, such that $\|\Sigma_2\| \leq \varepsilon / u_{\text{fp32}} \approx 2 \times 10^{-2}$
- Our idea: converting $U_2$ and $V_2$ to **single precision** only introduces an error of order $u_{\text{fp32}} \, \|\Sigma_2\| = \varepsilon$

# Error analysis

- Can use any number of precisions $u_1 \leq \varepsilon < u_2 < \ldots < u_p$

$$S_k = \left\{ i \leq r_\varepsilon : \frac{\varepsilon}{u_{k+1}} < \frac{\sigma_i}{\sigma_1} \leq \frac{\varepsilon}{u_k} \right\}, \quad k = 1\colon p$$

$$U_k \Sigma_k V_k^T = \sum_{i \in S_k} u_i \sigma_i v_i^T \quad \text{and} \quad \widehat{T} = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T$$

where $\widehat{U}_k$ and $\widehat{V}_k$ are stored in precision $u_k$.

# Error analysis

- Can use any number of precisions $u_1 \leq \varepsilon < u_2 < \ldots < u_p$

$$S_k = \left\{ i \leq r_\varepsilon : \frac{\varepsilon}{u_{k+1}} < \frac{\sigma_i}{\sigma_1} \leq \frac{\varepsilon}{u_k} \right\}, \quad k = 1 : p$$

$$U_k \Sigma_k V_k^T = \sum_{i \in S_k} u_i \sigma_i v_i^T \quad \text{and} \quad \widehat{T} = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T$$

where $\widehat{U}_k$ and $\widehat{V}_k$ are stored in precision $u_k$. Since for $k \geq 2$

$$\|U_k \Sigma_k V_k^T - \widehat{U}_k \Sigma_k \widehat{V}_k^T\| \leq (2u_k + u_k^2)\|\Sigma_k\| \leq (2 + u_k)\varepsilon\|A\|$$

we obtain

$$\boxed{\|A - \widehat{T}\| \leq \Big(2p - 1 + \sum_{k=2}^{p} u_k\Big)\varepsilon\|A\| = O(\varepsilon)\|A\|}$$

# Error analysis

- Can use any number of precisions $u_1 \le \varepsilon < u_2 < \ldots < u_p$

$$S_k = \left\{ i \le r_\varepsilon : \frac{\varepsilon}{u_{k+1}} < \frac{\sigma_i}{\sigma_1} \le \frac{\varepsilon}{u_k} \right\}, \quad k = 1 \colon p$$

$$U_k \Sigma_k V_k^T = \sum_{i \in S_k} u_i \sigma_i v_i^T \quad \text{and} \quad \widehat{T} = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T$$

where $\widehat{U}_k$ and $\widehat{V}_k$ are stored in precision $u_k$. Since for $k \ge 2$

$$\|U_k \Sigma_k V_k^T - \widehat{U}_k \Sigma_k \widehat{V}_k^T\| \le (2u_k + u_k^2)\|\Sigma_k\| \le (2 + u_k)\varepsilon\|A\|$$

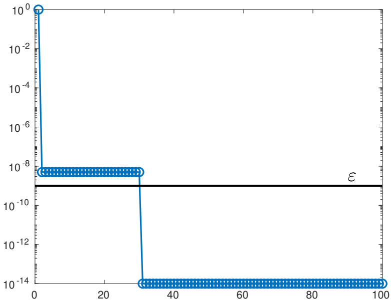
we obtain

$$\|A - \widehat{T}\| \le \Big(2p - 1 + \sum_{k=2}^{p} u_k\Big)\varepsilon\|A\| = O(\varepsilon)\|A\|$$

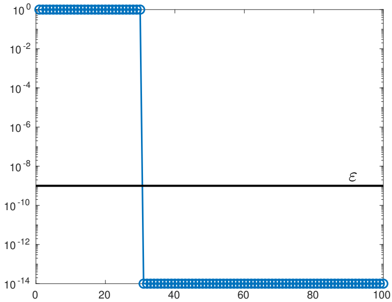
- Applicable to any low rank matrix $XY^T = \sum_{i=1}^{r_\varepsilon} x_i y_i^T$ with decaying $\|x_i y_i^T\|$. Example: $AP \approx Q_\varepsilon R_\varepsilon = Q_1 R_1 + \ldots + Q_p R_p$

# Examples of spectrum

Both matrices have $\varepsilon$-rank 30 (with $\varepsilon = 10^{-9}$) but present very different potential for mixed precision

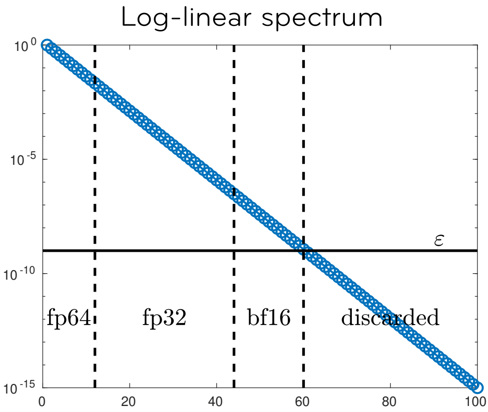



Large gain
(almost all in lower precision)



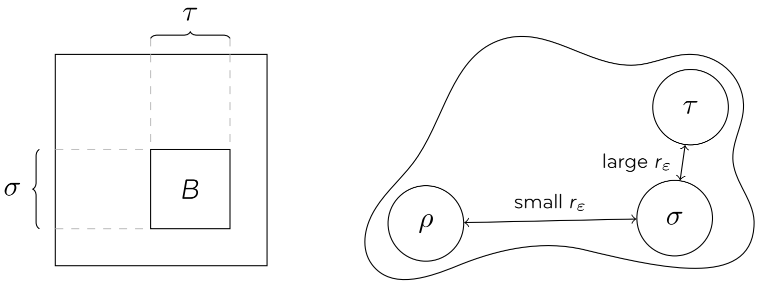

No gain
(all in higher precision)

# Examples of spectrum



# Data sparse matrices

- Data sparse matrices arise in several applications: BEM, PDEs, covariance matrices, ...

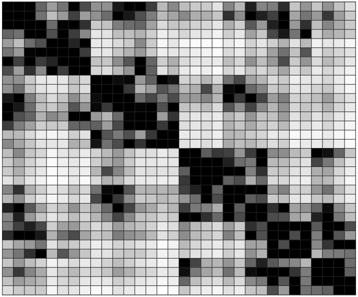



- They possess a block low rank structure: a block $B$ represents the interaction between two subdomains
  $\Rightarrow$ singular values decay rapidly for far away subdomains

$\Rightarrow$ High potential for mixed precision compression

# BLR matrices

BLR matrices (Amestoy et al.) use a flat 2D block partitioning

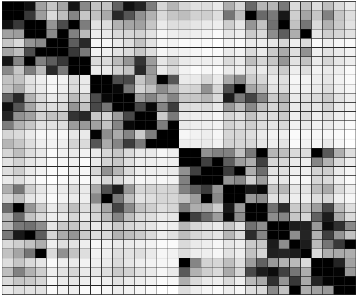


Example of a BLR matrix (Schur complement of a $64^3$ Poisson problem with block size $128$)

- Diagonal blocks are full rank
- Off-diagonal ones are stored in low rank form if their $\varepsilon$-rank is small enough
- $\varepsilon = 10^{-15} \rightarrow$ 50% entries kept

# BLR matrices

BLR matrices (Amestoy et al.) use a flat 2D block partitioning

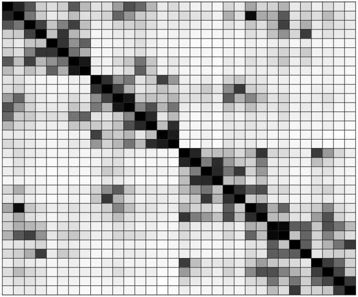


Example of a BLR matrix (Schur complement of a $64^3$ Poisson problem with block size $128$)

- Diagonal blocks are full rank
- Off-diagonal ones are stored in low rank form if their $\varepsilon$-rank is small enough
- $\varepsilon = 10^{-15} \rightarrow$ 50% entries kept
- $\varepsilon = 10^{-12} \rightarrow$ 36% entries kept

# BLR matrices

BLR matrices (Amestoy et al.) use a flat 2D block partitioning

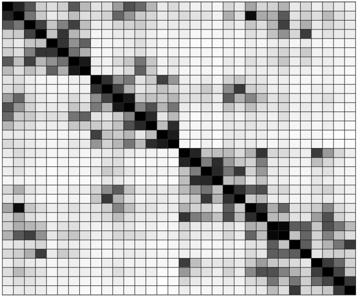


Example of a BLR matrix (Schur complement of a $64^3$ Poisson problem with block size $128$)

- Diagonal blocks are full rank
- Off-diagonal ones are stored in low rank form if their $\varepsilon$-rank is small enough
- $\varepsilon = 10^{-15} \to$ 50% entries kept
- $\varepsilon = 10^{-12} \to$ 36% entries kept
- $\varepsilon = 10^{-9} \to$ 23% entries kept

# BLR matrices

BLR matrices (Amestoy et al.) use a flat 2D block partitioning



Example of a BLR matrix (Schur complement of a $64^3$ Poisson problem with block size $128$)

- Diagonal blocks are full rank
- Off-diagonal ones are stored in low rank form if their $\varepsilon$-rank is small enough
- $\varepsilon = 10^{-15} \rightarrow$ 50% entries kept
- $\varepsilon = 10^{-12} \rightarrow$ 36% entries kept
- $\varepsilon = 10^{-9} \rightarrow$ 23% entries kept

Hierarchical data sparse matrices ($\mathcal{H}$, HSS, ...) not covered in this talk, but could also benefit from mixed precision

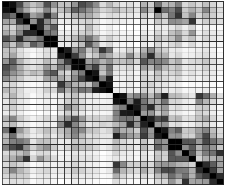
# Local vs global uniform precision compression

Should we approximate block $A_{ij} \approx T_{ij}$ such that

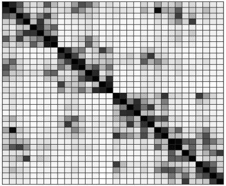
$$\|A_{ij} - T_{ij}\| \leq \varepsilon \, \|A_{ij}\| \quad \text{(local compression)}$$

$$\text{or} \quad \|A_{ij} - T_{ij}\| \leq \varepsilon \, \|A\| \quad \text{(global compression) ?}$$

- Global compression increases approximation error by a factor at most the number of block-rows/columns
- Generally worth the extra compression coming from blocks of norm less than $\|A\|$ (Higham & M., 2020)



Local compression
(38% entries kept)



Global compression
(23% entries kept)

# Local vs global mixed precision compression

- The set of singular vectors stored in precision $u_k$ for block $A_{ij} = U^{(ij)}\Sigma^{(ij)}V^{(ij)T}$ is

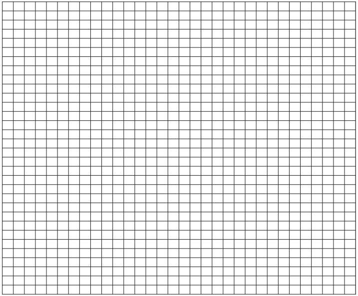
$$S_k^{(ij)} = \left\{ \ell \le r_\varepsilon : \frac{\varepsilon}{u_{k+1}} < \frac{\sigma_\ell^{(ij)}}{\sigma_1^{(ij)}} \le \frac{\varepsilon}{u_k} \right\} \quad \text{(local compression)}$$

$$S_k^{(ij)} = \left\{ \ell \le r_\varepsilon : \frac{\varepsilon}{u_{k+1}} < \frac{\sigma_\ell^{(ij)}}{\|A\|} \le \frac{\varepsilon}{u_k} \right\} \quad \text{(global compression)}$$

⇒ With global compression, $S_1$ may be empty for some blocks
Example: with double and single precisions, blocks such that $\|A_{ij}\| \le \varepsilon/u_{\mathrm{fp32}}\|A\|$ can be stored entirely in single precision

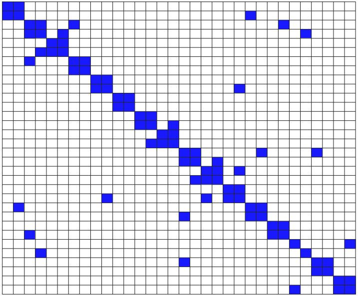
# Local vs global mixed precision compression

$(\varepsilon = 10^{-12})$



# Local vs global mixed precision compression

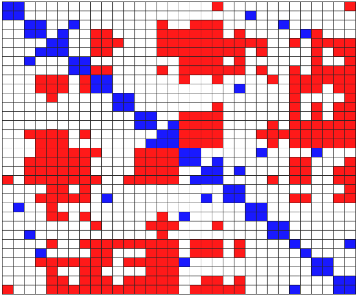
$(\varepsilon = 10^{-12})$



- Full rank blocks (**near field**) are in **double precision**

# Local vs global mixed precision compression

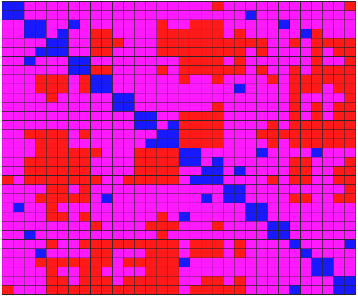
$(\varepsilon = 10^{-12})$



- Full rank blocks (**near field**) are in **double precision**
- **Far field** blocks are in **single precision**

# Local vs global mixed precision compression

$(\varepsilon = 10^{-12})$



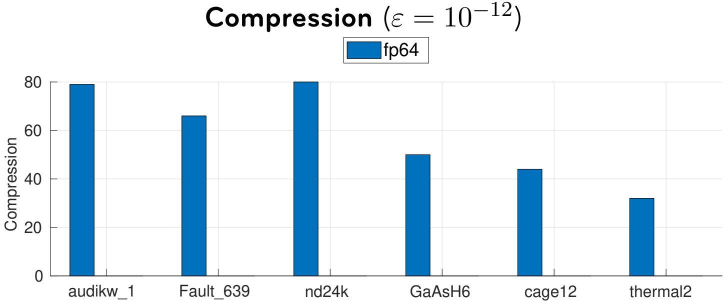
- Full rank blocks (**near field**) are in **double precision**
- **Far field** blocks are in **single precision**
- **Mid field** blocks are in **mixed precision**

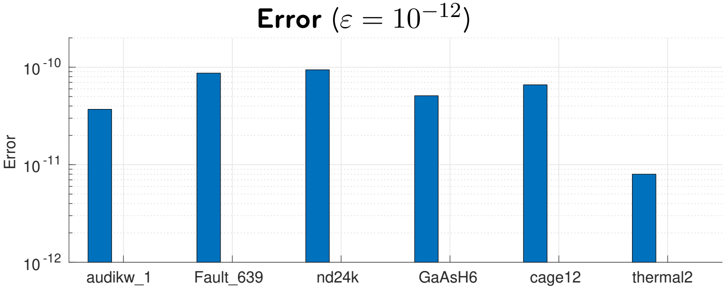
# Test matrices

- Dense matrices obtained from the root separator (Schur complement) of sparse matrices

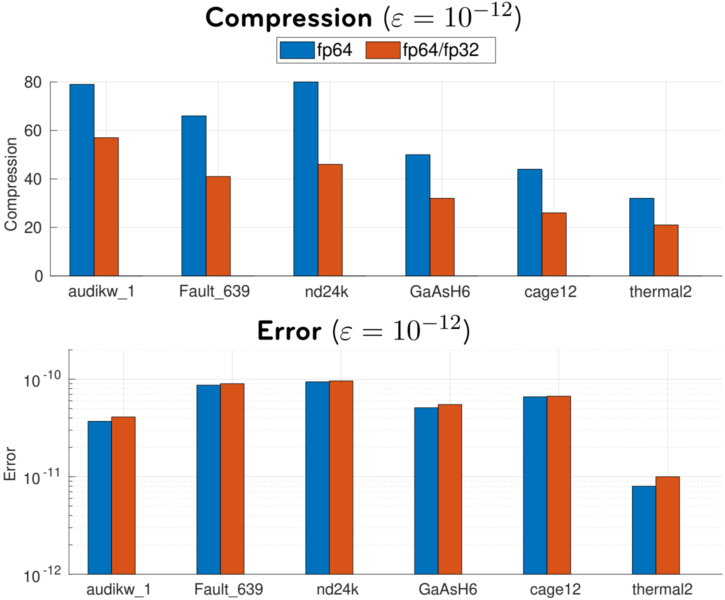
| Matrix | Application | $n$ |
|---|---|---|
| audikw_1 | Structural | $3768$ |
| Fault_639 | Structural | $7983$ |
| nd24k | 2D/3D | $7785$ |
| GaAsH6 | Chemistry | $6232$ |
| cage12 | Graph | $7323$ |
| thermal2 | Thermal | $1382$ |

- Block size is set to $128$


Compression ($\varepsilon = 10^{-12}$)
fp64
Compression
0
20
40
60
80
audikw_1
Fault_639
nd24k
GaAsH6
cage12
thermal2

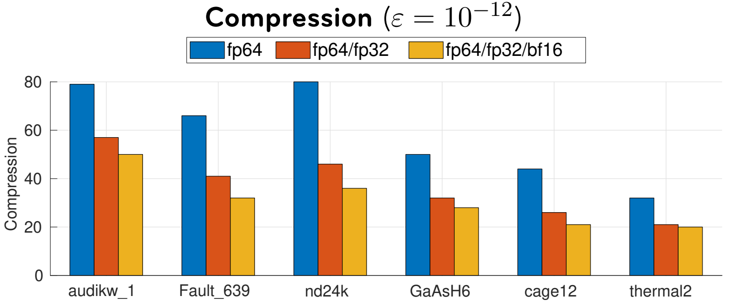

Error ($\varepsilon = 10^{-12}$)
Error
$10^{-10}$
$10^{-11}$
$10^{-12}$
audikw_1
Fault_639
nd24k
GaAsH6
cage12
thermal2

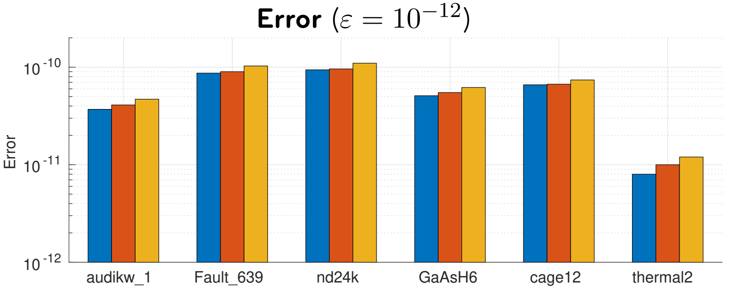
# Numerical results



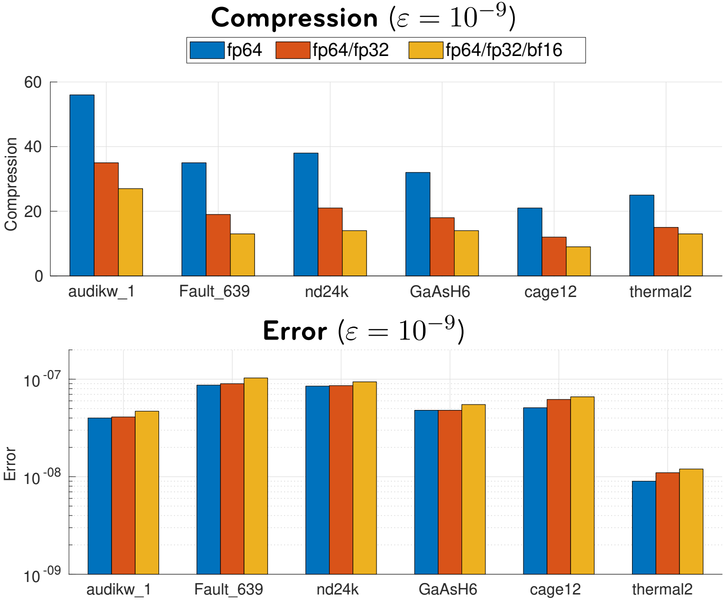

**Up to $1.7\times$ storage reduction with almost no error increase**

# Numerical results



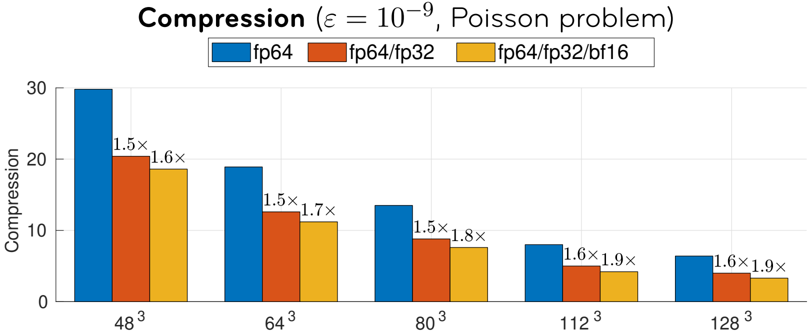




**Up to $2.2\times$ storage reduction with almost no error increase**




**Up to $2.7\times$ storage reduction with almost no error increase**

# Influence of problem size




**Gain due to mixed precision increases with problem size:**
$1.6\times$ (smallest) $\rightarrow$ $1.9\times$ (largest) storage reduction

# Mixed precision factorization of data sparse matrices

- Data sparse matrices can be factorized at a much lower cost than dense matrices
- Mixed precision can be used to further reduce this cost
- Example: a mixed precision low rank matrix $\widehat{T}$ can be multiplied with a vector $v$

$$\widehat{T}v = \left( \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T \right) v = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T v$$

  by computing $\widehat{U}_k \Sigma_k \widehat{V}_k v$ in precision $u_k$
- Other NLA operations can also be accelerated

# Mixed precision factorization of data sparse matrices

- Data sparse matrices can be factorized at a much lower cost than dense matrices
- Mixed precision can be used to further reduce this cost
- Example: a mixed precision low rank matrix $\widehat{T}$ can be multiplied with a vector $v$

$$\widehat{T}v = \left( \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T \right) v = \sum_{k=1}^{p} \widehat{U}_k \Sigma_k \widehat{V}_k^T v$$

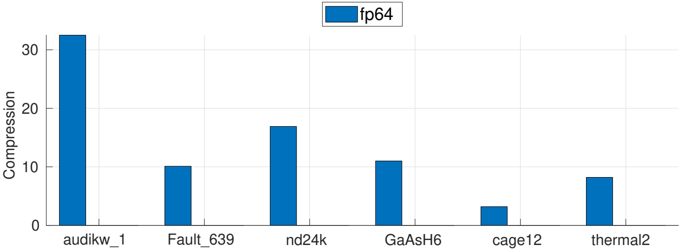
  by computing $\widehat{U}_k \Sigma_k \widehat{V}_k v$ in precision $u_k$
- Other NLA operations can also be accelerated
- Error analysis of BLR factorization in uniform precision $u$ (Higham and M., 2020) shows that

$$A + \Delta A = LU, \quad \|\Delta A\| \le c_1 \varepsilon \|A\| + c_2 u \|L\| \|U\|$$

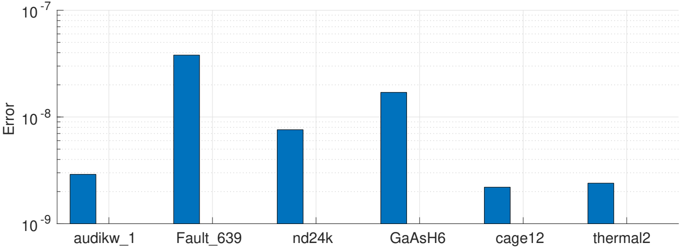
- Analysis can be generalized to mixed precision (ongoing work) with only a modest increase of $c_1$

# Preliminary flops results (no timings yet)

**Flops compression** ($\varepsilon = 10^{-9}$)

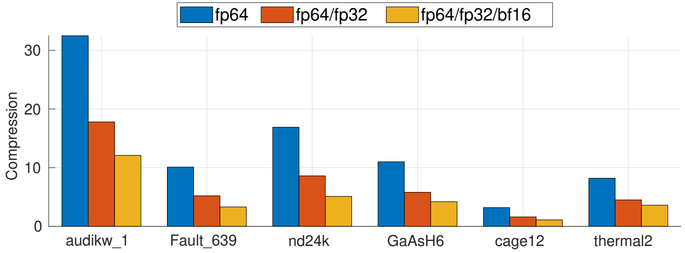



**Error** ($\varepsilon = 10^{-9}$)

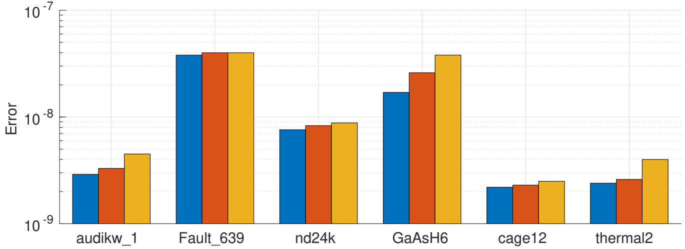


# Preliminary flops results (no timings yet)

**Flops compression** ($\varepsilon = 10^{-9}$)



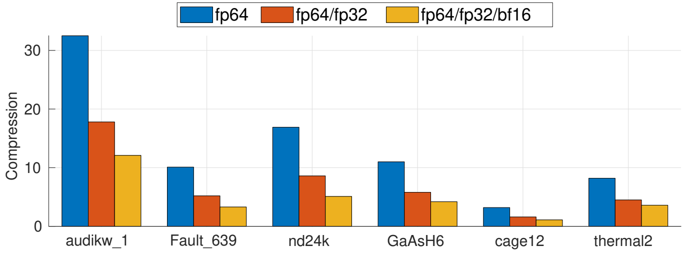

**Error** ($\varepsilon = 10^{-9}$)



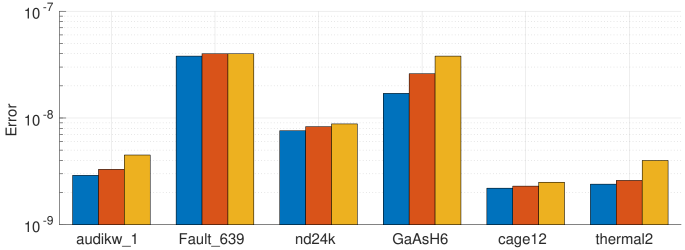

**Up to $3.3\times$ flops reduction with almost no error increase**

# Preliminary flops results (no timings yet)

**Flops compression** $(\varepsilon = 10^{-9})$



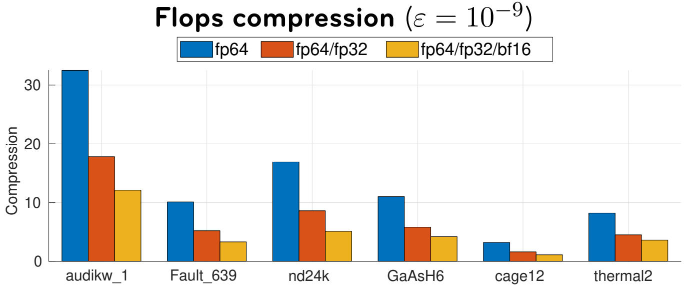

**Error** $(\varepsilon = 10^{-9})$

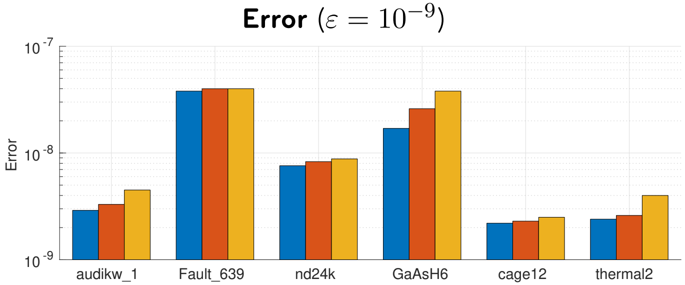



**Up to $3.3\times$ flops reduction with almost no error increase**

**$\rightarrow 3.3\times$ time reduction??**

# Preliminary flops results (no timings yet)




**Up to $3.3\times$ flops reduction with almost no error increase**

**$\rightarrow$ $3.3\times$ time reduction?? $7.0\times$ with GPU tensor cores**

# Conclusions

## Mixed precision SVD

- Given a matrix $A$ and a target accuracy $\varepsilon$, in what precision should we represent $A$?
- Naive answer: the lowest precision with unit roundoff less than $\varepsilon$
- Our answer: **it depends on its singular values!**

$\Rightarrow$ If rapidly decaying, precisions lower than $\varepsilon$ can be used

- Also applicable to QR and many other low rank decompositions

## Mixed precision compression of data sparse matrices

- Data sparse matrices are an ideal application due to their block low-rank structure
- Achieved up to $2.7\times$ storage reduction with fp64/fp32/bfloat16
- Can also accelerate factorization, up to $3.3\times$ flops reduction

$\Rightarrow$ Much work still needed to transform flops into time reduction!

# References (mixed precision algorithms)


- E. Carson and N. J. Higham. Accelerating the Solution of Linear Systems by Iterative Refinement in Three Precisions. *SIAM J. Sci. Comput.*, 40(2), A817–A847 (2018)
- P. Blanchard, N. J. Higham, and T. Mary. A Class of Fast and Accurate Summation Algorithms. *SIAM J. Sci. Comput.* 42(3), A1541–1557 (2020).
- P. Blanchard, N. J. Higham, F. Lopez, T. Mary, and S. Pranesh. Mixed Precision Block Fused Multiply-Add: Error Analysis and Application to GPU Tensor Cores. *SIAM J. Sci. Comput.* 42(3), C124–C141 (2020).
- F. Lopez and T. Mary. Mixed Precision LU Factorization on GPU Tensor Cores: Reducing Data Movement and Memory Footprint. MIMS EPrint 2020.20.
- A. Abdelfattah et al. A Survey of Numerical Methods Utilizing Mixed Precision Arithmetic. ArXiv:2007.06674 (2020).

# References (BLR matrices)


- P. R. Amestoy, C. Ashcraft, O. Boiteau, A. Buttari, J.-Y. L'Excellent, and C. Weisbecker. Improving Multifrontal Methods by Means of Block Low-Rank Representations *SIAM J. Sci. Comput.*, 37(3), A1451–A1474 (2015).
- P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. On the Complexity of the Block Low-Rank Multifrontal Factorization. *SIAM J. Sci. Comput.*, 39(4), A1710–A1740 (2017).
- P. R. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures. *ACM Trans. Math. Softw.*, 45(1), 2:1–2:26 (2019).
- N. J. Higham and T. Mary. Solving Block Low-Rank Linear Systems by LU Factorization is Numerically Stable. MIMS EPrint 2019.15.
- T. Mary. Block Low-Rank multifrontal solvers: complexity, performance, and scalability. PhD thesis (2017).