Block low-rank compression in mixed precision for the solution of sparse linear systems

Matthieu Gerest, Patrick Amestoy, Olivier Boiteau, Alfredo Buttari, Fabienne Jézéquel, Jean-Yves L'Excellent, Théo Mary

Block low-rank compression in mixed precision for the solution of sparse linear systems

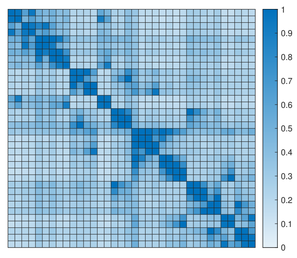
Matthieu Gerest, Cifre PhD student (EDF R&D, LIP6)
Joint work with P.Amestoy, O.Boiteau, A.Buttari, F.Jézéquel, J.-Y.L'Excellent, T.Mary

Low-precision arithmetics

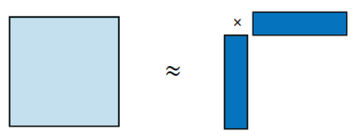
	Mantissa (m)	Exponent	Range	$u = 2^{-m-1}$
fp64 (double)	52 bits	11 bits	$10^{\pm 308}$	1×10^{-16}
fp32 (single)	23 bits	8 bits	$10^{\pm 38}$	6×10^{-8}
fp16 (half)	10 bits	5 bits	$10^{\pm 5}$	5×10^{-4}
bfloat16 (half)	7 bits	8 bits	$10^{\pm 38}$	4×10^{-3}

BLR matrices

We consider a certain class of matrices, whose off-diagonal blocks have low numerical ranks. More precisely, the singular values of such blocks decrease rapidly, typically following an exponential decay. BLR compression consists in approximating each of those block as a product of two smaller rectangular matrices (low-rank approximation). It may be based on a truncated SVD or QR decomposition.



Example of a BLR matrix
(*perf009*, RIS pumpfrom EDF).
Color scale: numerical ranks of the blocks

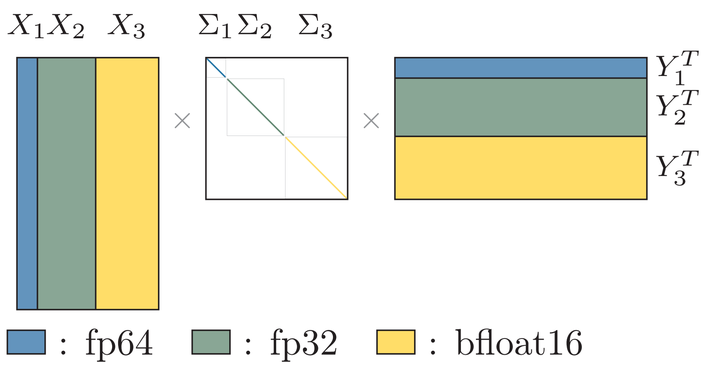


Low rank approximation of
a block: $B \approx X \times Y^T$

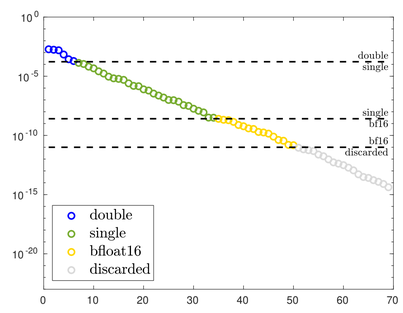
Low-rank approximation in mixed precision

We introduce a new approach to handle a low-rank approximation, in case it is based on a truncated SVD or QR decomposition. We propose to separate the columns into several groups, associated with different floating-point formats.

- A criterion for storing columns x_i and y_i in precision fp32: $\frac{\varepsilon}{u_{bf16}}\|B\| < \sigma_i \leq \frac{\varepsilon}{u_{fp32}}\|B\|$
- Compression error: $\|B - X\Sigma Y\| \lesssim 5\varepsilon\|B\|$, instead of $\varepsilon\|B\|$



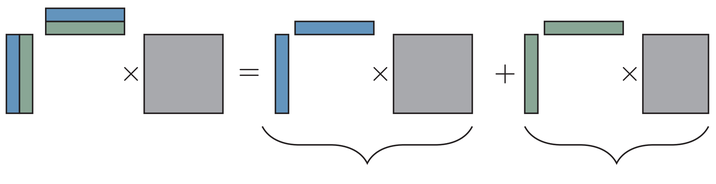
Approximation of a block as a truncated SVD



Repartition of the singular values
of a block: a typical example
(*perf009*, block (12,11))

LU factorization (dense matrices)

- Block LU factorization algorithm, step k:
  - → Compute $L_kU_k = A_{kk}$
  - → Update formula: for $i, j > k$,
    $A_{ij} \leftarrow A_{ij} - (A_{ik}U_k^{-1}) \times (L_k^{-1}A_{kj})$
- With BLR compression, the approximation $A_{ik} \approx X_{ik}Y_{ik}^T$ allows to reduce the number of operations.
- Example of kernel in mixed precision: multiplication LR × matrix :

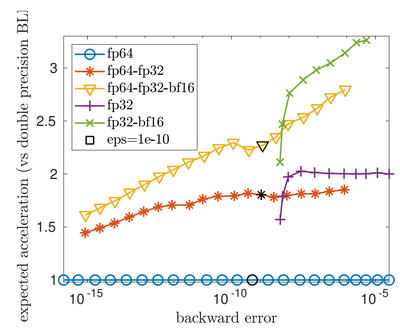


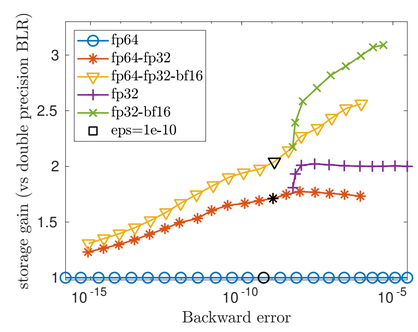
- This new algorithm is numerically stable (see [1]), like its monoprecision variant (see [2]):

$$\widehat{L}\widehat{U} = A + \Delta A, \quad \|\Delta A\| \leq \left(c_1\varepsilon + c_2\rho_n u_{fp64}\right)\|A\|$$

Results on dense matrices

- We emulate a LU factorization with BLR in 3 precisions: fp64, fp32 and bfloat16.
- Hypothesis: time cost = flops(fp64) $+\frac{1}{2}$flops(fp32) $+\frac{1}{4}$flops(bf16)
- We plot the relative gains with mixed precsion compared to double precision, as a function of the error. We notice that, for a given error, the mixed precision variant achieves better performances than the double precision ($\times 2$ to $\times 3$ in terms of storage and expected time).



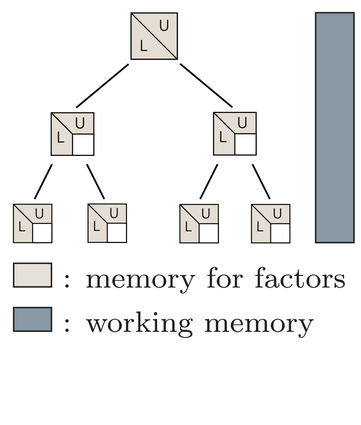


LU factorization of sparse matrices

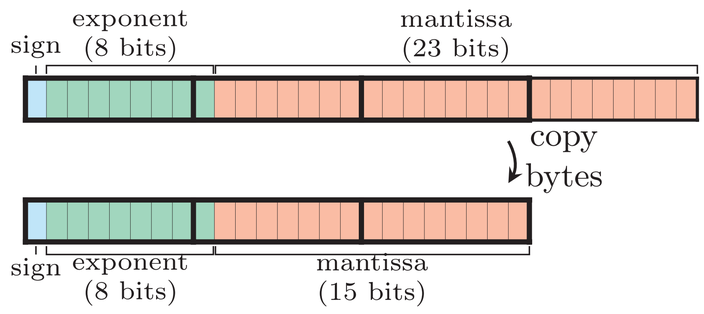
A multifrontal solver, such as MUMPS[3], computes a LU factorization of a sparse matrix. In order to do that, many partial LU factorizations of smaller dense matrices are computed.

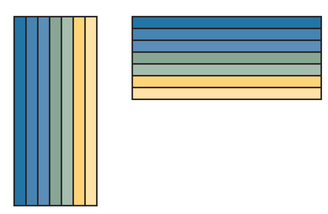
The LU factors are potentially BLR matrices, and their storage cost is a large part of the memory peak. We added an option in MUMPS that converts the low-rank blocks to mixed precision when they are not used.

If mixed-precision BLR is used for storage gains only, only a conversion operation is needed for the formats. Instead of the 3 common floating-point formats, we decided to use 7, respectively on 64, 56, 48, 40, 32, 24, and 16 bits.



- Example: conversion from fp32 to "fp24":





Representation of a low-rank block stored in 7 precisions

- Our first results show that, by adding mixed precision, there is a gain of storage between $\times 1.2$ and $\times 1.7$ regarding the LU factors:

Matrix	precision	Factor size (GBytes)	Memory peak (GBytes)	Scaled residual
thmgas	fp64	95	120	6.4E-14
	mixed	59	86	5.5E-14
perf009	fp64	25.6	36	1.3E-10
	mixed	20.5	32	1.4E-10

Perspectives

- Aim for times gains in MUMPS by performing computations in mixed precision
- Develop a variant of the algorithm that uses fp16 instead of bloat16. Scaling methoids will be required.
- A QR factorization algorithm may be accelerated using mixed precision

References

[1] P.Amestoy, O.Boiteau, A.Buttari, M.Gerest F.Jézéquel, J.-Y.L'Excellent,T.Mary. Mixed Precision Low Rank Approximations and their Application to Block Low Rank LU Factorization, 2021 (preprint)

[2] N.Higham, T.Mary, Solving Block Low-Rank Linear Systems by LU Factorization is Numerically Stable, IMA Journal of Numerical Analysis, 2019

[3] https://mumps-solver.org