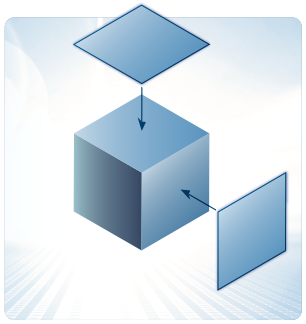
FIND OUT MORE AT https://hpl-mxp.org/

HPL-MxP

HPL-MxP MIXED-PRECISION BENCHMARK JUNE 2025

				CORES	HPL-MxP EFLOP/S	TOP 500 RANK	HPL Rmax EFLOP/S	SPEEDUP
1	EL CAPITAN	HPE Cray EX255a, AMD 4th Gen EPYC 24C 1.8GHz, AMD Instinct MI300A, Slingshot-11, TOSS	DOE/NNSA/LLNL USA	11,039,616	16.680	1	1.7420	9.6
2	AURORA	HPE Cray EX, Intel Exascale Compute Blade, Xeon CPU Max 9470 52C, 2.4 GHz, Intel GPU MAX, Slingshot-11	DOE/SC/ANL USA	8,159,232	11.643	3	1.0120	11.5
3	FRONTIER	HPE Cray EX235a, AMD Optimized 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11	DOE/SC/ORNL USA	8,560,640	11.390	2	1.3530	8.4
4	ABCI 3.0	HPE Cray XD670, Xeon Platinum 8558 48C 2.1GHz, NVIDIA H200 SXM5 141 GB, Infiniband NDR200	AIST JAPAN	479,232	2.363	15	0.1451	16.3
5	LUMI	HPE Cray EX235a, AMD Optimized 3rd Gen EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11	EuroHPC/CSC FINLAND	2,752,704	2.350	9	0.3797	6.2
6	FUGAKU	Fujitsu A64FX 48C 2.2GHz, Tofu D	RIKEN Center for Computational Science JAPAN	7,630,848	2.000	7	0.4420	4.5
7	LEONARDO	BullSequana XH2000, Xeon Platinum 8358 32C 2.6GHz, NVIDIA A100 SXM4 64 GB, Quad-rail NVIDIA HDR100 Infiniband	EuroHPC/CINECA ITALY	1,824,768	1.842	10	0.2412	7.6
8	TSUBAME4	HPE Cray XD665, AMD EPYC 9654 96C 3.55GHz, NVIDIA H100 SXM5 94Gb, Infiniband NDR200	CII, Institute of Science Tokyo JAPAN	172,800	0.641	46	0.0396	16.2
9	SELENE	NVIDIA DGX A100, AMD EPYC 7742 64C 2.25GHz, NVIDIA A100, Mellanox HDR Infiniband	NVIDIA USA	555,520	0.630	30	0.0635	9.9
10	PERLMUTTER	HPE Cray EX 235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-11	DOE/SC/LBNL/NERSC USA	888,832	0.590	25	0.0792	7.4

OVERVIEW



The HPL-MxP benchmark seeks to highlight the emerging convergence of high-performance computing (HPC) and artificial intelligence (AI) workloads. While traditional HPC focused on simulation runs for modeling phenomena in physics, chemistry, biology, and so on, the mathematical models that drive these computations require, for the most part, 64-bit accuracy. On the other hand, the machine learning methods that fuel advances in AI achieve desired results at 32-bit and even lower floating-point precision formats. This lesser demand for accuracy fueled a resurgence of interest in new hardware platforms that deliver a mix of unprecedented performance levels and energy savings to achieve the classification and recognition fidelity afforded by higher-accuracy formats.

HPL-MxP strives to unite these two realms by delivering a blend of modern algorithms and contemporary hardware while simultaneously connecting to the solver formulation of the decades-old HPL framework of benchmarking the largest supercomputing installations in the world. The solver method of choice is a combination of LU factorization and iterative refinement performed afterwards to bring the solution back to 64-bit accuracy. The innovation of HPL-MxP lies in dropping the requirement of 64-bit computation throughout the entire solution process and instead opting for low-precision (likely 16-bit) accuracy for LU, and a sophisticated iteration to recover the accuracy lost in factorization. The iterative method guaranteed to be numerically stable is the generalized minimal residual method (GMRES), which uses application of the L and U factors to serve as a preconditioner. The combination of these algorithms is demonstrably sufficient for high accuracy and may be implemented in a way that takes advantage of the current and upcoming devices for accelerating AI workloads.

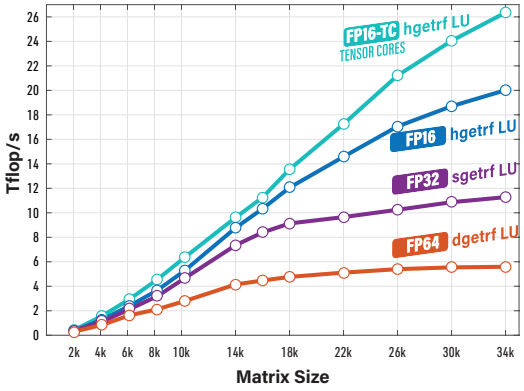
IN COLLABORATION WITH

CHARLES UNIVERSITY

MANCHESTER 1824 The University of Manchester

PERFORMANCE

Performance of various implementations of mixed-precision LU factorization routines



PUBLICATIONS

Jack Dongarra and Piotr Luszczek
HPL-MxP Benchmark: Mixed-Precision Algorithms, Iterative Refinement, and Scalable Data Generation The International Journal of High Performance Computing Applications, 2025. https://journals.sagepub.com/doi/abs/10.1177/10943420251382476

Piotr Luszczek *et al.*
Performance and Numerical Aspects of Decompositional Factorizations with FP64 Floating-Point Emulation in INT8 HPEC 2025. https://arxiv.org/abs/2509.23565

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 2019. http://eprints.maths.manchester.ac.uk/2733/

Azzam Haidar, Stanimire Tomov, Jack Dongarra, and Nicholas J. Higham
Harnessing GPU Tensor Cores for Fast FP16 Arithmetic to Speed up Mixed-Precision Iterative Refinement Solvers In Procedeedings of SC18, 2018. https://dl.acm.org/citation.cfm?id=3291719

Erin Carson and Nicholas J. Higham
Accelerating the Solution of Linear Systems by Iterative Refinement in Three Precisions SIAM J. SCI. COMPUT., Vol. 40, No. 2, pp. A817-A847, 2018. https://epubs.siam.org/doi/pdf/10.1137/17M1140819

Erin Carson and Nicholas J. Higham
A New Analysis of Iterative Refinement and its Application to Accurate Solution of Ill-Conditioned Sparse Linear Systems 2017. https://eprints.maths.manchester.ac.uk/2537/

Nicholas J. Higham, Srikara Pranesh, and Mawussi Zounon
Squeezing a Matrix into Half Precision, with an Application to Solving Linear Systems 2018. https://eprints.maths.manchester.ac.uk/2678/

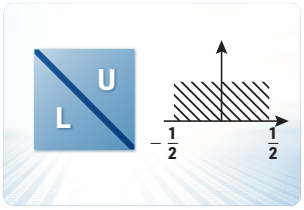
FIND OUT MORE AT https://hpl-mxp.org/

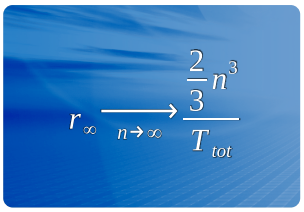
HPL-MxP

RULES

The idea for the benchmark is to solve a system of linear equations to 64-bit floating point accuracy by doing a mixed-precision factorization of a matrix and compute an approximate solution from the low-precision factorization (LU decomposition), and then use an iterative method like GMRES in 64-bit precision to iterate with the approximate low-precision solution to compute a final solution obtaining the accuracy one would have achieved by LU decomposition in 64-bit floating point arithmetic. The low-precision LU factors should be used as a preconditioner in the iterative algorithm.

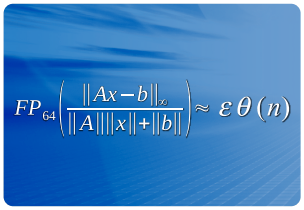
The benchmark should use the HPL benchmark harness (https://www.netlib.org/benchmark/hpl/) with a modification of the matrix generator. The generator will produce a non-symmetric matrix with the diagonal entries being the sum of the off-diagonal rows, this will force the matrix to be diagonally dominant.



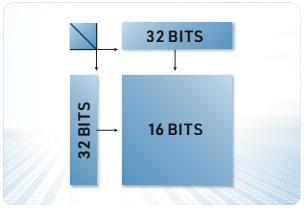


In an attempt to obtain uniformity across all computers in performance reporting, the algorithm used in solving the low-precision system of equations in the benchmark procedure must numerically conform to an LU factorization with partial pivoting. In particular, the operation count for the algorithm must be $\frac{2}{3}n^3 + O(n^2)$ double precision floating point operations even though double-precision arithmetic is not required.

The HPL harness computes a backward-error: $\frac{\|Ax-b\|_\infty}{\|A\|_\infty\|x\|_\infty+\|b\|_\infty}\times(n\times\epsilon)^{-1}$, where ϵ is the machine precision in 64-bit floating point arithmetic (on IEEE machines this is $\epsilon = 2^{-53}$) and n is the size of the problem. There is no restriction on the problem size.

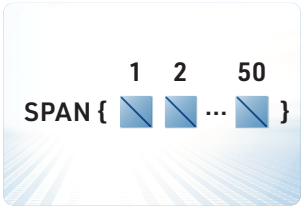


The implementation is allowed to do balancing to get the numbers within range of the floating point format, but the time to do the balancing must be included in the time to solution.

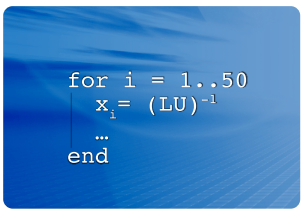


The factorization can use mixed precision during its construction, e.g., the panel factorization and triangular solves can be done in 32-bit arithmetic and the Schur complement (matrix-matrix multiply) can be computed in 16-bit arithmetic with 32-bit accumulation.

The computation rate is based on the time to solve the problem: factor the matrix in lower precision, perhaps balance the matrix to prevent overflow, perform GMRES in 64-bit floating point arithmetic using the LU factors as a preconditioner. If the implementation takes more than 50 iterations, the method should trigger a failure and the run is not valid.



In computing a rate of execution, $\frac{2}{3}n^3+\frac{3}{2}n^2$ operations ($\frac{2}{3}n^3-\frac{1}{2}n^2$accounts for LU factorization and $2n^2$ for the subsequent back- and forward-solves) will be divided by the complete time to solution to achieve operations per second.



As part of the submission of results we expect the submitter to provide a detailed explanation of the algorithm used in the submission.

We have provided a reference implementation whose purpose is to show how the benchmark could be implemented. We do not expect this to be used in actually running of the benchmark. Optimizations should be applied to achieve higher performance than the reference implementation could achieve. The reference implementation can be found here on Bitbucket (https://bitbucket.org/icl/hpl-ai/)