MAGMA

Matrix Algebra on GPU and Multicore Architectures

Matrix Algebra on GPU and Multicore Architectures (MAGMA) is a collection of next-generation linear algebra libraries for heterogeneous architectures. MAGMA is designed and implemented by the team that developed LAPACK and ScaLAPACK, incorporating the latest developments in hybrid synchronization-avoiding and communication-avoiding algorithms, as well as dynamic runtime systems. Interfaces for the current LAPACK and BLAS standards are supported to enable computational scientists to seamlessly port any linear algebra-reliant software components to heterogeneous architectures. MAGMA allows applications to fully exploit the power of current heterogeneous systems of multi/many-core CPUs and multiple GPUs to deliver the fastest possible time to accurate solution within given energy constraints.

FIND OUT MORE AT

https://icl.utk.edu/magma

HYBRID ALGORITHMS

MAGMA uses a hybridization methodology, where algorithms of interest are split into tasks of varying granularity, and their execution is scheduled over the available hardware components. Scheduling can be static or dynamic. In either case, small non-parallelizable tasks, often on the critical path, are scheduled on the CPU, and larger more parallelizable ones, often Level-3 BLAS, are scheduled on the GPU.

PERFORMANCE & ENERGY EFFICIENCY

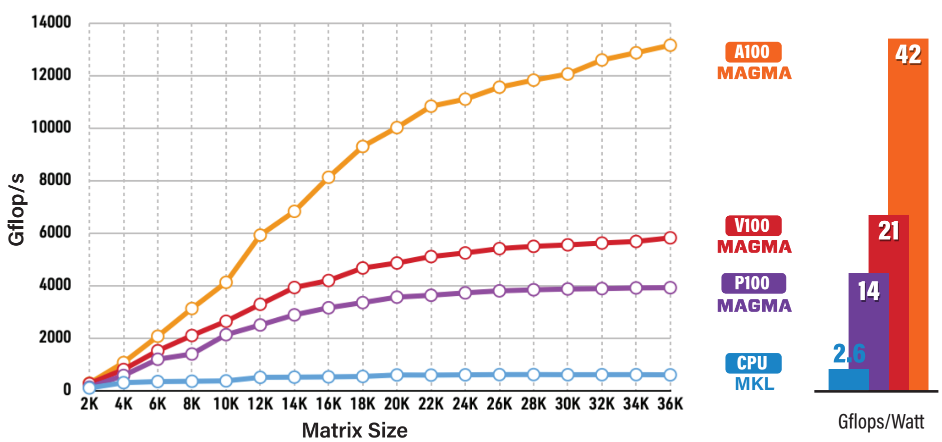
MAGMA LU factorization in double-precision arithmetic

CPU Intel Xeon E5-2650 v3 (Haswell), 2 x 10 cores @ 2.30 GHz

P100 NVIDIA Pascal GPU, 56 SMs x 64 @ 1.19 GHz

V100 NVIDIA Volta GPU, 80 SMs x 64 @ 1.37 GHz

A100 NVIDIA Ampere GPU, 108 SMs x 64 @ 1.41 GHz



FEATURES AND SUPPORT

- **MAGMA 2.5.4** FOR CUDA
- **clMAGMA 1.4** FOR OpenCL
- **MAGMA MIC 1.4** FOR Intel Xeon Phi
- **hipMAGMA 2.0** FOR HIP

CUDA	OpenCL	Intel Xeon Phi	HIP	
●	●	●	●	Linear system solvers
●	●	●	●	Eigenvalue problem solvers
●	●		●	Auxiliary BLAS
●			●	Batched LA
●		●		Sparse LA
●	●	●	●	CPU/GPU Interface
●	●	●	●	Multiple precision support
●			●	Mixed precision (including FP16)
●			●	Non-GPU-resident factorizations
●			●	GPU-only factorizations
●	●	●	●	Multicore and multi-GPU support
●				MAGMA Analytics/DNN v1.2
●	●	●	●	LAPACK testing
●	●	●	●	Linux
●	●			Windows
●	●			macOS

INDUSTRY COLLABORATION

NVIDIA

Long-term collaboration and support on the development of MAGMA.

Intel Parallel Computing Center

The objective of the Innovative Computing Laboratory's IPCC is the development and optimization of numerical linear algebra libraries and technologies for applications, while tackling current challenges in heterogeneous Intel® Xeon Phi™ coprocessor-based High Performance Computing.

AMD

Long-term collaboration and support on the development of clMAGMA, the OpenCL™ port of MAGMA, and hipMAGMA.

IN COLLABORATION WITH

Berkeley University of California, University of Colorado Denver, Inria, KAUST

SPONSORED BY

U.S. Department of Defense, U.S. Department of Energy, ECP, National Science Foundation

WITH SUPPORT FROM

AMD, Intel, MathWorks, NVIDIA

MAGMA

Matrix Algebra on GPU and Multicore Architectures

MAGMA BATCHED

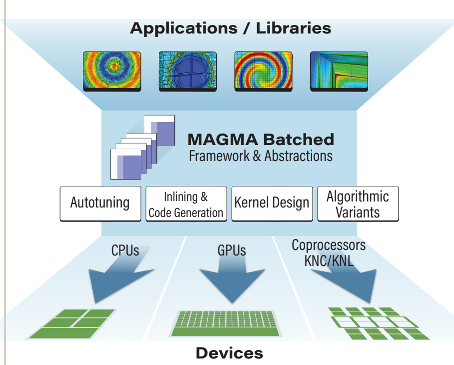
Batched factorization of a set of small matrices in parallel

Numerous applications require factorization of many small matrices:

- Deep learning
- Structural mechanics
- Astrophysics
- Sparse direct solvers
- High-order FEM simulations

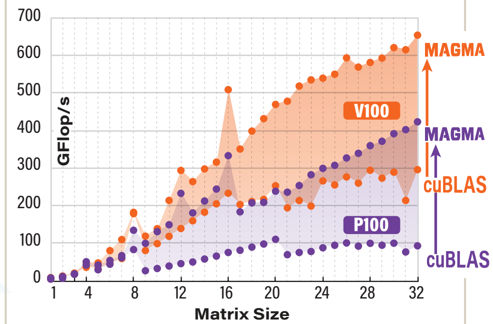
ROUTINES

- ✓ LU, QR, and Cholesky
- ✓ Solvers and matrix inversion
- ✓ All BLAS 3 (fixed + variable)
- ✓ SYMV, GEMV (fixed + variable)



PERFORMANCE OF BATCHED LU

in double-precision arithmetic on 1 million matrices



MAGMA 2.5.4 Driver Routines

	MATRIX	OPERATION	ROUTINE	INTERFACES CPU	INTERFACES GPU
LINEAR EQUATIONS	GE	Solve using LU	{sdcz}gesv	✓	✓
		Solve using MP	{zc,ds}gesv		✓
	SPD/HPD	Solve using Cholesky	{sdcz}posv	✓	✓
		Solve using MP	{zc,ds}posv		✓
LEAST SQUARES	GE	Solve LLS using QR	{sdcz}gels		✓
		Solve using MP	{zc,ds}geqrsv		✓
STANDARD EVP	GE	Compute e-values,	{sdcz}geev	✓	
		optionally e-vectors		✓	
	SY/HE	Computes all e-values,	{sd}syevd	✓	✓
		optionally e-vectors	{cz}heevd	✓	✓
		Range (D&C)	{cz}heevdx		✓
		Range (B&I It.)	{cz}heevx	✓	✓
		Range (MRRR)	{cz}heevr	✓	✓
STAND. SVP	GE	Compute SVD,	{sdcz}gesvd	✓	
		optionally s-vectors	{sdcz}gesdd	✓	
GENERALIZED EVP	SPD/HPD	Compute all e-values,	{sd}sygvd	✓	
		optionally e-vectors	{cz}hegvd	✓	
		Range (D&C)	{cz}hegvdx	✓	
		Range (B&I It.)	{cz}hegvx	✓	
		Range (MRRR)	{cz}hegvr	✓	

Abbreviations

GE	General
SPD/HPD	Symmetric/Hermitian Positive Definite
TR	Triangular
D&C	Divide & Conquer
B&I It	Bisection & Inverse Iteration
MP	Mixed-Precision Iterative Refinement

Naming Convention

magma_{routine name}[_gpu]

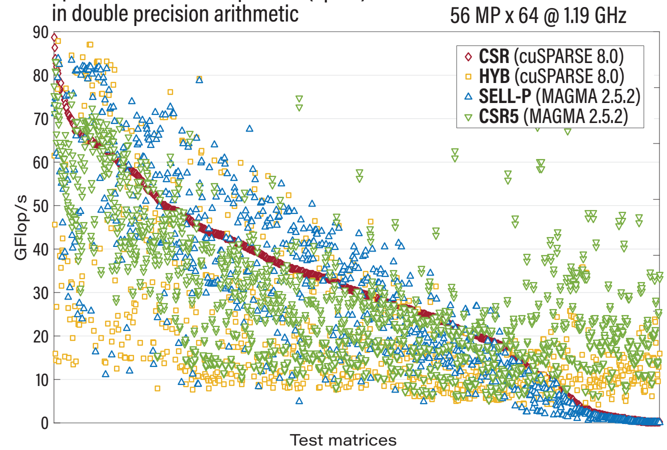
MAGMA SPARSE

ROUTINES	BiCG, BiCGSTAB, Block-Asynchronous Jacobi, CG, CGS, GMRES, IDR, Iterative refinement, LOBPCG, LSQR, QMR, TFQMR
PRECONDITIONERS	ILU / IC, Jacobi, ParILU, ParILUT, Block Jacobi, ISAI
KERNELS	SpMV, SpMM
DATA FORMATS	CSR, ELL, SELL-P, CSR5, HYB

PERFORMANCE

Sparse matrix - vector product (SpMV) in double precision arithmetic

NVIDIA Pascal P100 GPU
56 MP x 64 @ 1.19 GHz



from The University of Florida Sparse Matrix Collection
http://www.cise.ufl.edu/research/sparse/matrices/

IN COLLABORATION WITH

Berkeley University of California
University of Colorado Denver
Inria
KAUST

SPONSORED BY

U.S. Department of Defense
U.S. Department of Energy
ECP
National Science Foundation

WITH SUPPORT FROM

AMD
intel
MathWorks
NVIDIA

ICL INNOVATIVE COMPUTING LABORATORY
The University of Tennessee Knoxville