MAGMA

MATRIX ALGEBRA ON GPU AND MULTICORE ARCHITECTURES

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

HYBRID ALGORITHMS

MAGMA uses a hybridization methodology where algorithms of interest are split into tasks of varying granularity and their execution 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.

MAGMA BLAS

MAGMA BLAS targets a subset of BLAS routines for NVIDIA GPUs that are specific to MAGMA and can improve on CUBLAS. MAGMA BLAS supports streaming and includes routines for Fermi and for the older generation of Tesla GPUs. Also included is a new ZGEMM obtained through autotuning.

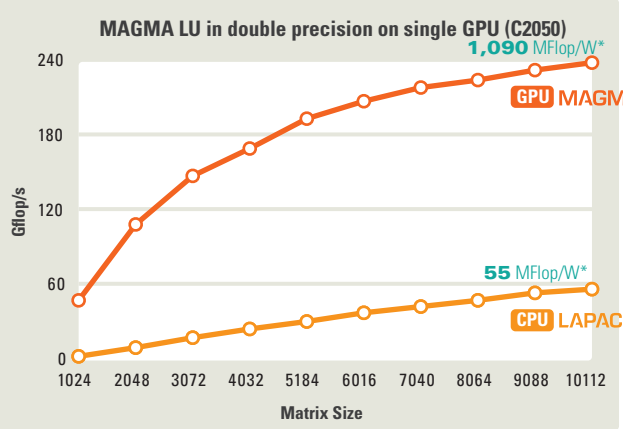
MAGMA 1.1

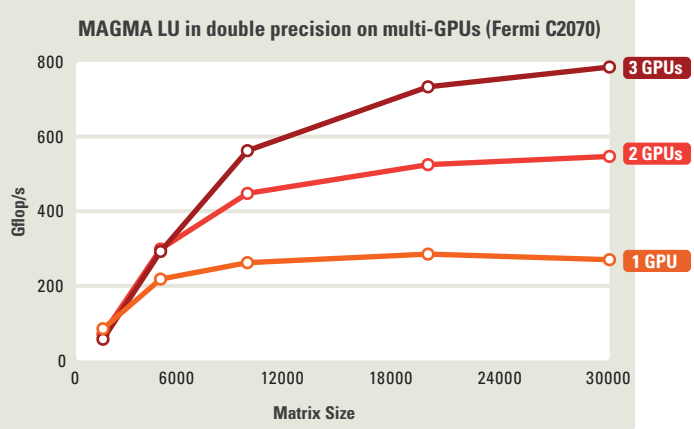
- Linear Systems Solvers
- Eigenvalue Problem Solvers
- MAGMA BLAS
- CPU and GPU Interfaces
- Multiple Precision Support (S/D/C/Z, including mixed precision)
- *NEW* Non-GPU-resident one-sided factorizations
- *NEW* Autotuned ZGEMM
- *NEW* Multicore and multi-GPU Support
- *NEW* Tile one-sided factorizations w/ StarPU dynamic scheduling
- *NEW* Matrix inversion
- *NEW* LAPACK testing
- Linux, Windows, Mac OS

UPCOMING

- OpenCL port
- Distributed systems support
- Autotuning framework
- Extended functionality, including sparse linear algebra

PERFORMANCE





GPU Fermi C2050 (448 CUDA Cores @ 1.15 GHz)
+ Intel Q9300 (4 cores @ 2.50 GHz)
DP peak **515** + **40** GFlop/s
Power * ~**220** W

CPU AMD Istanbul
[8 sockets x 6 cores (48 cores) @2.8GHz]
DP peak **538** GFlop/s
Power * ~**1,022** W

Keeneland system, using one node
3 NVIDIA GPUs (M2070 @ 1.1 GHz, 5.4 GB)
2 x 6 Intel Cores (X5660 @ 2.8 GHz, 23 GB)

* Computation consumed power rate (total system rate minus idle rate), measured with KILL A WATT PS, Model P430

FIND OUT MORE AT http://icl.eecs.utk.edu/magma/

A COLLABORATION OF

The University of Tennessee · Berkeley · University of Colorado Denver · Inria · KAUST

INNOVATIVE COMPUTING LABORATORY
The University of Tennessee

WITH SUPPORT FROM

MathWorks · Microsoft · NVIDIA

SPONSORED BY

CITR
CENTER FOR INFORMATION TECHNOLOGY RESEARCH

MAGMA

MAGMA 1.1 ROUTINES & FUNCTIONALITIES	SINGLE GPU	MULTI-GPU STATIC	MULTI-GPU DYNAMIC
One-sided Factorizations (LU, QR, Cholesky)	✓	✓	✓
Linear System Solvers	✓		✓
Linear Least Squares (LLS) Solvers	✓		✓
Matrix Inversion	✓		✓
Singular Value Problem (SVP)	✓		
Non-symmetric Eigenvalue Problem	✓		
Symmetric Eigenvalue Problem	✓		
Generalized Symmetric Eigenvalue Problem	✓		

SINGLE GPU	Hybrid LAPACK algorithms with static scheduling and LAPACK data layout
MULTI-GPU STATIC	Hybrid LAPACK algorithms with 1D block cyclic static scheduling and LAPACK data layout
MULTI-GPU DYNAMIC	Tile algorithms with StarPU scheduling and tile matrix layout

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]

DRIVER ROUTINES IN MAGMA 1.1

	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		✓
LLS	GE	Solve LLS using QR	{sdcz}geqrs		✓
		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		✓	
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	✓	

COMPUTATIONAL ROUTINES IN MAGMA 1.1

	MATRIX	OPERATION	ROUTINE	INTERFACES CPU	INTERFACES GPU
LINEAR EQUATIONS	GE	LU	{sdcz}getrf	✓	✓
		Solve	{sdcz}getrs		✓
		Invert	{sdcz}getri		✓
	SPD/HPD	Cholesky	{sdcz}potrf	✓	✓
		Solve	{sdcz}potrs		✓
		Invert	{sdcz}potri		✓
	TR	Invert	{sdcz}trtri	✓	
ORTHOGONAL FACTORIZATIONS	GE	QR	{sdcz}geqrf	✓	
		Generate Q	{sd}orgqr	✓	✓
			{cz}ungqr	✓	✓
		Multiply matrix by Q	{sd}ormqr	✓	✓
			{cz}unmqr	✓	✓
	GE	LQ factorization	{sdcz}gelqf	✓	✓
		QL factorization	{sdcz}geqlf	✓	
		Multiply matrix by Q	{sd}ormql	✓	✓
			{cz}unmql	✓	✓
STANDARD EVP	GE	Hessenberg reduction	{sdcz}gehrd	✓	
		Generate Q	{sd}orghr	✓	
			{cz}unghr	✓	
	SY/HE	Tridiagonalization	{sd}sytrd	✓	
			{cz}hetrd	✓	
		Generate Q	{sd}orgtr		✓
			{cz}ungtr		✓
		Multiply by Q	{sd}ormtr	✓	✓
			{cz}unmtr	✓	✓
SVD	GE	Bidiagonalization	{sdcz}gebrd	✓	
GENER-ALIZED EVP	SPD/HPD	Reduction to standard form	{sd}sygst	✓	✓
			{cz}hegst	✓	✓

FIND OUT MORE AT http://icl.eecs.utk.edu/magma/

A COLLABORATION OF

INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE

WITH SUPPORT FROM

SPONSORED BY

CITR CENTER FOR INFORMATION TECHNOLOGY RESEARCH