Matrix Algebra on GPU and Multicore Architectures (MAGMA)

PART II : Hands-on training

Stan Tomov

Innovative Computing Laboratory
Department of Computer Science
University of Tennessee, Knoxville

CSCS-USI Tutorial on GPU-enabled numerical libraries
Lugano, Switzerland
September 14-15, 2013

Outline

- Install MAGMA
- Using MAGMA
  - Documentation, naming conventions, and functionality
- Solving a linear system of equations

  $$Ax = b$$
- Solving an eigenvalue problem

  $$Ax = \lambda x$$
- DGEMM example
- Writing a hybrid algorithm

Install MAGMA

- **Get MAGMA 1.4**
  > wget http://icl.cs.utk.edu/projectsfiles/magma/downloads/magma-1.4.0.tar.gz

- **Unpack the library**
  > tar zxvf magma-1.4.0.tar.gz

- **Prerequisites: LAPACK, BLAS, and CUDA toolkit**
  > module load cudatoolkit intel

- **Modify 'make.inc' (where are CUDA & LAPACK, and for what GPU)**
  See examples in make.inc.{mkl-gcc | mkl-icc | mkl-ilp64 | mkl-shared | acml | atlas | goto | shared},
  > cp make.inc.mkl-gcc make.inc

  and modify make.inc by setting GPU_TARGET, MKLROOT, and CUDADIR, e.g.,

  ```
  GPU_TARGET ?= Kepler
  MKLROOT     = $(INTEL_PATH)/mkl
  CUDADIR     = $(CRAY_CUDATOOLKIT_DIR)
  ```

- **Create libmagma.a in 'lib' and testing drivers in 'testing'**
  > make

- **For more information on installation, read file 'README'**

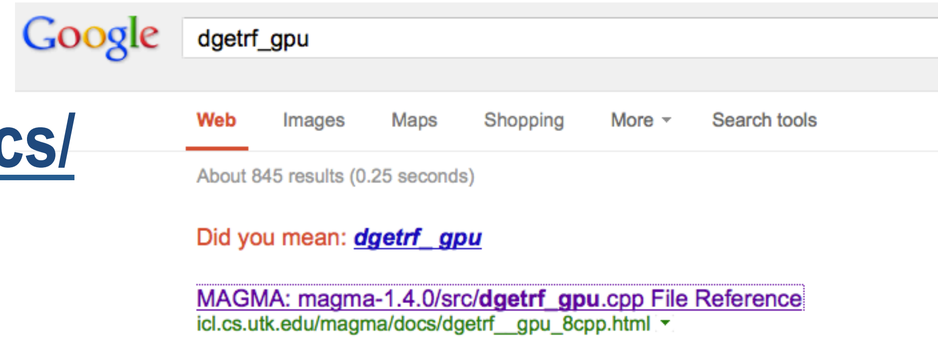
Using MAGMA

- **Support is provided through the MAGMA user forum** http://icl.cs.utk.edu/magma/forum/viewforum.php?f=2

NaN errors with dpotrf and dpotrf_gpu by fletchjp » Tue Dec 28, 2010 7:08 pm	9	3095	by fletchjp Thu Sep 12, 2013 12:03 pm
GMRES on magma by nitinb60 » Sun Aug 12, 2012 8:15 pm	4	1064	by fletchjp Thu Sep 12, 2013 11:02 am
Help please with choice of forums by fletchjp » Sat Apr 13, 2013 2:44 pm	4	1628	by fletchjp Thu Sep 12, 2013 10:42 am
the error when I compile magma1.4.0 in vs2010 by Eva Joo » Mon Sep 09, 2013 4:57 am	2	62	by Eva Joo Thu Sep 12, 2013 4:54 am
Undefined reference to cuda functions within libmagma by Matt Phillips » Mon Sep 09, 2013 10:40 pm	1	63	by Matt Phillips Mon Sep 09, 2013 11:17 pm
crash testing strsv, using OpenBLAS by Matt Phillips » Thu Sep 05, 2013 9:15 pm	0	101	by Matt Phillips Thu Sep 05, 2013 9:15 pm
MAGMA Installation: CLAPACK reference BLAS problem by psrivas2 » Tue Sep 03, 2013 4:07 am	0	84	by psrivas2 Tue Sep 03, 2013 4:07 am
Running MAGMA across several GPUs on several nodes by hsahasra » Tue Aug 13, 2013 1:32 pm	1	310	by mgates3 Fri Aug 30, 2013 3:48 pm
magma_init/finalize missing in fortran interface by stachon » Wed Aug 28, 2013 4:02 am	1	116	by mgates3 Fri Aug 30, 2013 2:13 pm
Error: BLAS/LAPACK routine 'magma_' gave error code -7 by christianHEL » Thu Aug 22, 2013 2:37 pm	2	197	by christianHEL Fri Aug 23, 2013 3:50 pm
Problems testing dsyevd with magma-1.4.0 by dougrabson » Thu Aug 15, 2013 5:44 am	1	173	by mgates3 Wed Aug 21, 2013 2:35 pm

Using MAGMA

- **Doxygen documentation**
  http://icl.cs.utk.edu/magma/docs/



Using MAGMA

- **Naming conventions (e.g., magma_dgesv_gpu)**
  - Prefix **magma_** or **magmablas_**
  - Followed by precision
    s single, **d** double, **c** single complex, or **z** double complex
    ds mixed double-single, or **zc** double complex-single complex
  - Matrix type
    general **sy**mmetric **he**rmetian **po**sitive definite
    orthogonal **un**itary **tr**iangular
  - Operation

sv	solve	**ev**	eigenvalue problem
trf	triangular factorization	**gv**	generalized eigenvalue
qrf	QR factorization		

  - Suffix _**gpu** if input and output are on the GPU memory
    (no suffix indicates CPU interface – use GPUs to accelerate where input and output are on the CPU)

ICL UT INNOVATIVE COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE
Department of Electrical Engineering and Computer Science

Using MAGMA

- **MAGMA functionality**
  http://icl.cs.utk.edu/graphics/posters/files/SC12-MAGMA.pdf

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 1.3

	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 1.3

	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	✓	
		QR w/ pivoting	{sdcz}geqp3		✓
		Generate Q	{sd}orgqr	✓	✓
			{cz}ungqr	✓	✓
		Multiply matrix by Q	{sd}ormqr	✓	✓
			{cz}unmqr	✓	✓
		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	{sd}sygst	✓	✓
		form	{cz}hegst	✓	✓

artment of Electrical Engineering
and Computer Science

Examples / Exercises

- **Solving a linear system of equations**

  Ax = b

```
#include "magma.h"
#include "magma_lapack.h"
…

magma_init();
cublasInit();
…
double *hA, *hB;                                                    // A is a typical array on the CPU
magma_malloc_pinned((void **)&hA,  lda * N * sizeof(double));      // A can be allocated in pinned memory
magma_malloc_cpu(    (void **)&hB, nrhs* M *sizeof(double));
…
init_matrix(M, N, hA, lda);
…
magma_dgetrf( M, N, hA, lda, ipiv, &info);                          // LU factorization (using CPU+GPU)
lapackf77_dgetrs( MagmaNoTransStr, &N, &nrhs,                       // Solve on CPU with LAPACK
                  hA, &lda, ipiv, hB, &ldb, &info );
```

Alternatively, there is a direct MAGMA function to solve:
see **testing_dgesv.cpp** and **testing_dgetrf.cpp**

Examples / Exercises

- **Solving an eigenvalue problem**

$$Ax = \lambda x$$

```
#include "magma.h"
#include "magma_lapack.h"
…

magma_init();
cublasInit();
…
double *hA;                                                          // A is a typical array on the CPU
magma_malloc_pinned((void **)&hA,  lda * N * sizeof(double));    // A can be allocated in pinned memory
…
init_matrix(M, N, hA, lda);
…
magma_dsyevd( opts.jobz, opts.uplo,
              N, hA, lda, w1,
              h_work, lwork,
              iwork, liwork, &info );
```

see **testing_dsyevd.cpp**

Examples / Exercises

- **DGEMM example**

```
salloc -N1 --gres=gpu:1 --time=00:30:00
module load cudatoolkit intel
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/apps/dom/intel/composer_xe_2013/mkl/lib/intel64/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/apps/opcode/CUDA-5.0/lib64
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/apps/dom/intel/composer_xe_2013.1.117/compiler/lib/intel64/
```

aprun -n1 -N1 -e OMP_NUM_THREADS=16 -d16 ./testing_dgemm -l –c

```
MAGMA 1.4.0 , capability 3.0
device 0: Tesla K20X, 732.0 MHz clock, 5759.6 MB memory, capability 3.5

Usage: ./testing_dgemm [options] [-h|--help]

transA = N, transB = N
    M      N      K   MAGMA Gflop/s (ms)  CUBLAS Gflop/s (ms)   CPU Gflop/s (ms)  MAGMA error  CUBLAS error
===========================================================================================================
1088  1088  1088    612.88 (   4.20)       969.12 (   2.66)         3.13 ( 823.44)       6.92e-15        6.92e-15
2112   2112  2112    667.85 (  28.21)     1100.09 (  17.13)        6.79 (2775.89)       8.12e-15        7.95e-15
3136  3136  3136    675.16 (  91.36)     1144.59 (  53.89)       11.14 (5538.38)        1.13e-14        1.13e-14
…
```

see file **testing_dgemm.cpp, sgemm.pdf, dgemm_fermi.cu**, and directory **testing**

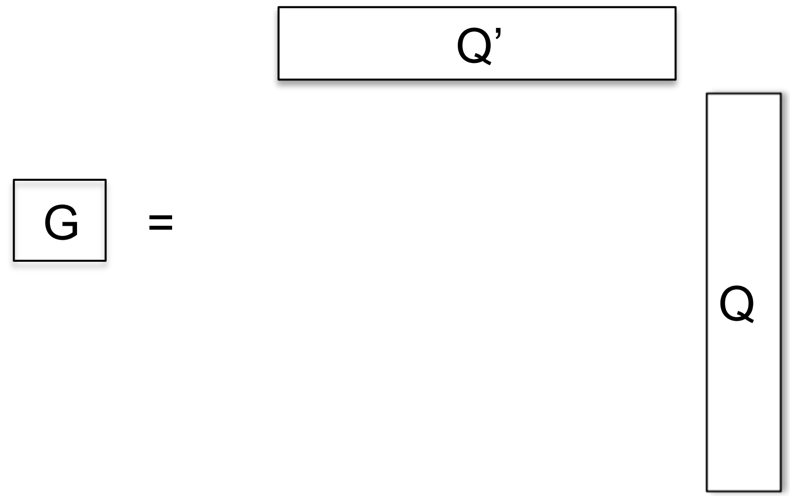
Examples / Exercises

- **Writing a hybrid algorithm**

Develop a hybrid CPU-GPU algorithm for this Matlab code

```
function [Q,R] = chol_qr_it(A)
   i=0;
   cn = 200;
   Q = A;
   G = Q'*Q;
   n = size(A,2);
   R = eye(n);

   while cn > 100, i = i + 1
       [u,s,v]=svd(G);
       [q,r]=qr(sqrt(s)*v');
       R = r * R;
       cn = sqrt(cond(s));
       Q = Q * inv(r);
       if cn>100
          G = Q'*Q;
       end;
   end;
return
```



Computations on small data to be done on the CPU

Computations on large data to be done on the GPU

Examples / Exercises

- **Writing a hybrid algorithm**

  Develop a hybrid CPU-GPU algorithm for this Matlab code

```
function [Q,R] = chol_qr_it(A)
  i=0;
  cn = 200;
  Q = A;
  G = Q'*Q;                                  cublasSgemm( ... );       // G = Q' *Q (GPU computation)
  n = size(A,2);
  R = eye(n);

  while cn > 100, i = i + 1                  cublasGetVector( ... );   // G -> work (send the result G to the CPU)
     [u,s,v]=svd(G);                         sgesvd_( ... );           //  svd G on the CPU
     [q,r]=qr(sqrt(s)*v');                   sgeqrf_( ... );           //  QR on the CPU
     R = r * R;
     cn = sqrt(cond(s));                     cublasSetVector( ... );   // send r back to the the GPU)
     Q = Q * inv(r);                         cublasStrsm( ... );       // Triangular matrix solve on the GPU
     if cn>100
        G = Q'*Q;
     end;
  end;
return
```

see **testing_dgegqr_gpu.cpp** and **dgegqr_gpu.cpp**

Contributions

The MAGMA program style follows the general guidelines for Sca/LAPACK in terms of interfaces, copyrights and licensing, citing the authors of the software, and documentation:
http://www.netlib.org/lapack-dev/lapack-coding/program-style.html

- Routine Naming and Design
  - Use BLAS
  - Develop in double complex; other precisions are generated
  - Machine parameters are determined at **magma_init** (as needed for tuning)
  - Block sizes are provided by **extern "C" int magma_get_ROUTINE_nb(int m)**, i.e., not by ilaenv
- Language, source formatting, timing, and testing
  - Written in C/C++; wrappers for Fortran are supported
- See file **'ContributorsGuige.txt'**

Collaborators and Support

MAGMA team

http://icl.cs.utk.edu/magma

PLASMA team

http://icl.cs.utk.edu/plasma

Collaborating partners

University of Tennessee, Knoxville
University of California, Berkeley
University of Colorado, Denver
INRIA, France (StarPU team)
KAUST, Saudi Arabia

NVIDIA Microsoft The MathWorks intel AMD NSF U.S. DEPARTMENT OF ENERGY