CUBLAS
Sharan Chetlur, 19 May 2016

NVIDIA
CUBLAS : HIGH PERFORMANCE BLAS FOR NVIDIA GPUs

Introduction

Full implementation of the BLAS standard

Plus a few extensions for batched and reduced precision routines, and one or two other things (e.g. SYRKX, GEAM)

Vanilla cuBLAS : All data and compute on the GPU

cuBLAS-XT : Data on either host or GPU, scales across GPUs in a node

Free with CUDA toolkit, built and maintained by NVIDIA

Several external collaborators
CUBLAS: > 5 TFLOPS SINGLE PRECISION

- cuBLAS 8.0 RC on M40, Base clocks
- cuBLAS 7.5 on K40, Base clocks
- Input and output data on device
- m=n=k=4096, transpose=no, side=right, fill=lower

Performance may vary based on OS and software versions, and motherboard configuration.
Performance may vary based on OS and software versions, and motherboard configuration.

- cuBLAS 8.0 RC on M40, Base clocks, Input and output data on device
- cuBLAS 7.5 on K40, Base clocks, Input and output data on device
- \( m=n=k=4096 \)
CUBLAS-XT: >9 TFLOPS ON A SINGLE NODE

Multi-GPU Performance Scaling

- cuBLAS 8.0 RC on K80, base clocks, input and output data on device
- cuBLAS 7.5 on K80, base clocks, input and output data on device

Performance may vary based on OS and software versions, and motherboard configuration.
REPRODUCIBILITY

Run-to-run consistency

Bitwise identical results across runs

Given toolkit/driver versions, hardware and problem

User opt-in for non-reproducible execution

SYMV has a faster atomic path

Integral to our internal QA process

Necessary for debugging large, concurrent codes
REDUCED PRECISION

cuBLAS 8.0

New cublasHgemm() routine performs computation in FP16 format

FP16 vector arithmetic also available via intrinsics
  
  Compiler does not automatically vectorize
  
  Not all math.h functions

Evaluating other library improvements

  cuSPARSE support for fp16 iterative solver refinement iterations

  cuFFT direct fp16 transforms
MIXED PRECISION

“Ex” routines

SGEMMEx/SSYRKEx - FP16/int8 I/O, FP32 math

CGEMMEx/CHERKEx - int8 I/O, FP32 math

cublasStatus_t cublasSgemmEx (handle, transa, transb, m, n, k, 
const float *alpha, /* host or device pointer */
const void *A, 
cudaDataType Atype, lda, 
const void *B, 
cudaDataType Btype, ldb, 
const float *beta, /* host or device pointer */
void *C, 
cudaDataType Ctype, ldc);

Generic GEMM - custom I/O and Math types
BATCHED ROUTINES
BATCH_FIXED equivalent

cublasStatus_t cublasSgemmBatched (handle, transa, transb, m, n, k,
    const float *alpha, /* host or device pointer */
    const float *Aarray[], lda,
    const float *Barray[], ldb,
    const float *beta, /* host or device pointer */
    float *Carray[],
    int ldc,
    int batchCount);

Exists for S, D, C, Z GEMM variants
BATCHED ROUTINES

Strided batched GEMM

cublasStatus_t cublasSgemmStridedBatched (handle, transa, transb, m, n, k,
const float *alpha,
const float *A, lda, long long int strideA,
const float *B, ldb, long long int strideB,
const float *beta,
float *C, ldc, long long int strideC,
int batchCount);

Exists for S, D, C, Z, H GEMM variants