ACCELERATED LIBRARIES

Jeff Larkin, December 04, 2018
3 WAYS TO ACCELERATE APPLICATIONS

- Libraries
  - “Drop-in” Acceleration
- Compiler Directives
  - Easily Accelerate Applications
- Programming Languages
  - Maximum Flexibility
LIBRARIES: EASY, HIGH-QUALITY ACCELERATION

EASE OF USE
Using libraries enables GPU acceleration without in-depth knowledge of GPU programming

“DROP-IN”
Many GPU-accelerated libraries follow standard APIs, thus enabling acceleration with minimal code changes

QUALITY
Libraries offer high-quality implementations of functions encountered in a broad range of applications

PERFORMANCE
NVIDIA libraries are tuned by experts
GPU ACCELERATED LIBRARIES
“Drop-in” Acceleration for Your Applications

DEEP LEARNING
- cuDNN
- TensorRT
- DALI

SIGNAL & IMAGE PROCESSING
- cuFFT
- NVIDIA NPP
- nvJPEG

LINEAR ALGEBRA
- cuBLAS
- CUTLASS
- cuSOLVER
- CUDA Math API
- cuSPARSE
- cuRAND
- cuSOLVER

PARALLEL ALGORITHMS
- nvGRAPH
- NCCL
- Thrust
# MATH LIBRARIES 9.2 - HIGHLIGHTS

## VOLTA PLATFORM SUPPORT
- Volta architecture optimized GEMMs, & GEMM extensions for Volta Tensor Cores (cuBLAS)
- Out-of-box performance on Volta (all libraries)

## NEW ALGORITHMS
- Mixed-precision Batched GEMM for attention models (cuBLAS)
- Image Augmentation and batched image processing routines (NPP)
- Batched pentadiagonal solver (cuSPARSE)

## PERFORMANCE
- GEMM optimizations for RNNs (cuBLAS)
- Faster image processing (NPP)
- Prime factor FFT performance (cuFFT)
- SpMV performance (cuSPARSE)

## MEMORY & FOOTPRINT OPTIMIZATION
- Large FFT sizes on multi-GPU systems (cuFFT)
- Modular functional blocks with small footprint (NPP)
CUDA 10.0 - MATH LIBRARIES

**TURING**
- Turing optimized GEMMs, & GEMM extensions for Tensor Cores
- Turing architecture-optimized libraries

**PERFORMANCE**
- Large FFT & 16-GPU Strong Scaling
- Symmetric Eigensolver & Cholesky Performance
- cuSPARSE Sparse-Dense Matrix Multiply Performance

**NEW ALGORITHMS AND APIS**
- GPU-accelerated hybrid JPEG decoding
- FP16 & INT8 GEMMs for TensorRT Inference

**COMPATIBILITY & RELEASE CADENCE**
- Faster & Independent Library Releases
- Library and CUDA compatibility with enterprise drivers
cuBLAS
GPU-accelerated library for dense linear algebra

Accelerated library with complete BLAS plus extensions

- Supports all 152 standard routines for single, double, complex, and double complex
- Supports half-precision (FP16), integer (INT8) matrix and mixed precision multiplication operations
- Batched routines for higher performance on small problem sizes
- Host and device-callable interface
- XT interface supports distributed computations across multiple GPUs

https://developer.nvidia.com/cublas
cuBLAS 9.2

GEMM Performance for DL RNNs and Convolutional Seq2Seq Models

GEMM performance for

- Small tile sizes used in RNN models, Convolutional Seq2seq and OpenAI
- No API changes

Volta architecture optimized heuristics and kernels

API and Error Logging for debug and traceability

5-8x Tensor Op speedup on key DL sizes

https://developer.nvidia.com/cublas
cuBLAS 10.0
Optimized GEMM Performance for Deep Learning

- Turing optimized GEMMs & GEMM extensions for Tensor Cores
- GEMM Performance Tuned for sizes used in various DL models
- API and Error Logging for debug and traceability

https://developer.nvidia.com/cublas
cuBLAS 10.1: New Matrix Multiplication API

Offers future-proofing, flexibility and auto-tuning for GEMM routines

Abstraction of BLAS extensions and special data layouts used in DL applications

- Mixed-precision GEMMs, architecture specific data layouts, complex input/output formats

Supports custom configs & layouts (i.e. workspace) to perform intermediate calculations

Enables auto-tuning efforts with Find-GEMM API to pick optimal algo for given problem sizes
Next: Faster multi-GPU BLAS

cuBLASXT Roadmap: Dense matrix multiplication

\[ D = \alpha A \times B + \beta C \]
CUTLASS 1.0

Collection of CUDA C++ templates for linear algebra computations

- Thread-wide, warp-wide, block-wide, device-wide

Extensive support for mixed-precision GEMM computations

- Floating point (16-bit, 32-bit, 64-bit)
- Integer (8-bit)
- Volta Tensor Cores using WMMA API

Up to 90% of cuBLAS performance on certain GEMMs

Open Source: github.com/NVIDIA/cutlass
CUTLASS 1.1
High-performance Matrix Multiplication in Open Source CUDA C++

- Turing optimized GEMMs
  - Integer (8-bit, 4-bit and 1-bit) using WMMA
- Batched strided GEMM
- Support for CUDA 10.0
- Updates to documentation and more examples

https://github.com/NVIDIA/cutlass
CUTLASS 1.1
High-performance Matrix Multiplication in Open Source CUDA C++

- Turing optimized GEMMs
  - Integer (8-bit, 4-bit and 1-bit) using WMMA
- Batched strided GEMM
- Support for CUDA 10.0
- Updates to documentation and more examples

https://github.com/NVIDIA/cutlass

CUTLASS operations reach 90% of CUBLAS Performance

DGEMM | HGEMM | IGEMM | SGEMM | WMMA (F16) | WMMA (F32)
---|---|---|---|---|---
cuFFT
Complete Fast Fourier Transforms Library

Complete Multi-Dimensional FFT Library

“Drop-in” replacement for CPU FFTW library

Real and complex, single- and double-precision data types

Includes 1D, 2D and 3D batched transforms

Support for half-precision (FP16) data types

Supports flexible input and output data layouts

XT interface now supports up to 8 GPUs

https://developer.nvidia.com/cufft
cuFFT 9.2

Performance and Memory optimizations

Prime-factor FFT performance with fused Bluestein kernels

Support for large datasets with new API for low memory usage

Static library without callbacks for datacenter deployments

<table>
<thead>
<tr>
<th>Dimension</th>
<th># of GPUs w/ 16GB RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D</td>
<td>1</td>
</tr>
<tr>
<td>&lt;2^30 (1.0E+9)</td>
<td>&lt;2^30 (1.0E+9)</td>
</tr>
<tr>
<td>2D</td>
<td>&lt;2^15 (32.6k)</td>
</tr>
<tr>
<td>3D</td>
<td>&lt;2^11 (1.3k)</td>
</tr>
</tbody>
</table>

Faster Image & Signal Processing

* V100 and CUDA 9 (r384); Intel Xeon Broadwell, dual socket, E5-2698 v4@ 2.6GHz, 3.5GHz Turbo with Ubuntu 14.04.5 x86_64 with 128GB System Memory
* P100 and CUDA 8 (r361); For cublas CUDA 8 (r361); Intel Xeon Haswell, single-socket, 16-core E5-2698 v3@ 2.3GHz, 3.6GHz Turbo with CentOS 7.2 x86-64 with 128GB System Memory

https://developer.nvidia.com/cufft
cuFFT 10.0
Multi-GPU Scaling across DGX-2 and HGX-2

Strong scaling across 16-GPU systems - DGX-2 and HGX-2

Multi-GPU R2C and C2R support

Large FFT models across 16-GPUs

Upto 17TF performance on 16-GPUs - 3D 1K FFT

https://developer.nvidia.com/cufft
cuFFT 10.0
Multi-GPU Scaling across DGX-2 and HGX-2

- Strong scaling across 16-GPU systems - DGX-2 and HGX-2
- Multi-GPU R2C and C2R support
- Large FFT models across 16-GPUs - effective 512GB vs 32GB capacity

https://developer.nvidia.com/cufft
cuSPARSE
Sparse Linear Algebra on GPUs

Optimized Sparse Matrix Library

- Optimized sparse linear algebra BLAS routines for matrix-vector, matrix-matrix, triangular solve
- Support for variety of formats (CSR, COO, block variants)
- Incomplete-LU and Cholesky preconditioners
- Support for half-precision (fp16) sparse matrix-vector operations

https://developer.nvidia.com/cusparse
cuSPARSE 9.2

Sparse GEMM Performance for Scientific Computing and Deep Learning

- Mergepath SpMV Speedup
- New csrsm2 triangular solver
- Batched pentadiagonal solver
- Bug-fixes and Performance Optimizations

10x SpMV speedup on select matrices

[Graph showing speedup comparison]

*cuSPARSE 9.2 Vs cuSPARSE 9.1 comparison on V100*
cuSOLVER 9.2

Dense Solver Performance Improvements for Scientific Computing

Bug-fixes & Performance Optimizations

Enhancements:
- Zero-free diagonal reordering for sparse matrix factorization
- METIS matrix reordering option

Dense Solver Performance - 40% Faster

- cuSOLVER 9.2 on V100, Driver r396
- cuSOLVER 8 on P100, Driver r361
- Host system: Supermicro E5-2698 v4@2.20GHz 3.6GHz Turbo (Broadwell) HT On
- CentOS 7.2 x86-64 with 128GB System Memory

https://developer.nvidia.com/cublas
cuSOLVER
Linear Solver Library

Library for Dense and Sparse Direct Solvers

Supports Dense Cholesky, LU, (batched) QR, SVD and Eigenvalue solvers

Sparse direct solvers & Eigen solvers

Includes a sparse refactorization solver for solving sequences of matrices with a shared sparsity pattern

Used in a variety of applications such as circuit simulation and computational fluid dynamics

Sample Applications
• Computer Vision
• CFD
• Newton’s method
• Chemical Kinetics
• Chemistry
• ODEs
• Circuit Simulation

https://developer.nvidia.com/cusolver
cuSOLVER 10.0

Dense Linear Algebra Performance for Scientific Computing

Accelerated Performance for:
- Cholesky factorization (*POTRF):
  \[ [A] = [L][L]^T \]
- Symmetric eigensolver (*SYEVD):
  \[ [A][v_i] = \lambda_i[v_i] \]
- Generalized symmetric eigensolver (*SYGVD):
  \[ [A][v_i] = \lambda_i[B][v_i] \]

- cuSOLVER 9.2 on V100, Driver r396
- cuSOLVER 8 on P100, Driver r361
- Host system: Supermicro E5-2698 v4@2.20GHz 3.6GHz Turbo (Broadwell) HT On
- CentOS 7.2 x86-64 with 128GB System Memory

https://developer.nvidia.com/cublas
cuSOLVER 10: Dense Linear Algebra

Eigen solver & Cholesky Optimizations

10.0: Improved performance with new implementations for
- Cholesky factorization
- Symmetric & Generalized Symmetric eigensolver
- QR factorization

Future: continue perf tuning & add new functionality:
- Selective eigenvalue/vector solvers
- SVD Perf and batch APIs
- Un-symmetric Eigensolvers

Benchmarks use 2 x Intel Gold 6140 (Skylake) processors and NVIDIA GV100 (Volta) GPUs
Library Compatibility & Release Cadence
Monthly Releases, Independent Versioning & Driver Compatibility

Planned monthly release of libraries
- To support release agility needs from DL and HPC customers

Independent Library Versioning
- Industry standard semantic versioning (Major.Minor.Patch)

Compatibility across 2 LTS driver versions
Math Libraries 10.0 - Summary

**TURING TENSOR CORE 2.0**
- Turing optimized GEMMs, & GEMM extensions for Tensor Cores 2.0 (cuBLAS, CUTLASS)
- Out-of-box performance on Turing (all libraries)

**PERFORMANCE**
- Large FFT & 16-GPU Perf Scaling on DGX-2/HGX-2 (cuFFT)
- FP16 & INT8 GEMM perf for DL inference (cuBLAS)
- Symmetric Eigensolver & Cholesky Perf (cuSOLVER)

**NEW ALGORITHMS & APIs**
- GPU-accelerated hybrid JPEG decoding (nvJPEG)
- New Mat-mul and GEMM Find APIs (cuBLAS)
- Mixed-precision batched GEMV, GEMM for Complex data types (cuBLAS)

**COMPATIBILITY & RELEASE CADENCE**
- Faster & Independent Library Releases (starting w/ cuBLAS in Oct, others to follow)
- Single library compatible across N and N-1 LTS drivers (r410 and r384)