



ArrayFire Webinar:
OpenCL and CUDA Trade-Offs and Comparisons

GPU Software Features

Programmability

Portability

Scalability

Performance

Community

Leading GPU Computing Platforms

Programmability

Portability



Scalability

Performance



Community

Performance

- Both CUDA and OpenCL are fast
- Both *can* fully utilize the hardware
- Devil in the details:
 - Hardware type, algorithm type, code quality



Performance

- ArrayFire results at end of webinar



Scalability

- Laptops --> Single GPU Machine
 - Both CUDA and OpenCL scale, no code change
- Single GPU Machine --> Multi-GPU Machine
 - User managed, low-level synchronization
- Multi-GPU Machine --> Cluster
 - MPI



Scalability

Interesting developments:

- Memory Transfer Optimizations
 - CUDA GPUDirect technology
- Mobile GPU Computing
 - OpenCL available on ARM, Imgttec, Freescale, ...



Scalability



- Laptops --> Single GPU Machine
 - ArrayFire's JIT optimizes for GPU type
- Single GPU Machine --> Multi-GPU Machine
 - ArrayFire's *deviceset()* function is super easy
- Multi-GPU Machine --> Cluster
 - MPI



Portability

- CUDA is NVIDIA-only
 - Open source announcement
 - Does not provide CPU fallback
- OpenCL is the open industry standard
 - Runs on AMD, Intel, and NVIDIA
 - Provides CPU fallback



Portability



- ArrayFire is *fully* portable
 - Same ArrayFire code runs on CUDA or OpenCL
 - Simply select the right library



Community

- NVIDIA CUDA Forums – 26,893 topics
- AMD OpenCL Forums – 4,038 topics
- Stackoverflow CUDA Tag – 1,709 tags
- Stackoverflow OpenCL Tag – 564 tags



Community



- AccelerEyes GPU Forums – 1,435 topics
 - Largest GPU forums by a software company
- Next largest
 - PGI GPU Forums – 485 topics



Programmability

- Both CUDA and OpenCL are low-level
 - Time consuming kernel development
 - Data-parallel algorithm design
- Focus on programmability interfaces



Programmability



Faster

Slower

SSE or
AVX

Time-consuming

Easy-to-use

Programmability



Faster

Writing
Kernels

Slower

SSE or
AVX

Time-consuming

Easy-to-use

Programmability



Faster

Writing
Kernels

Slower

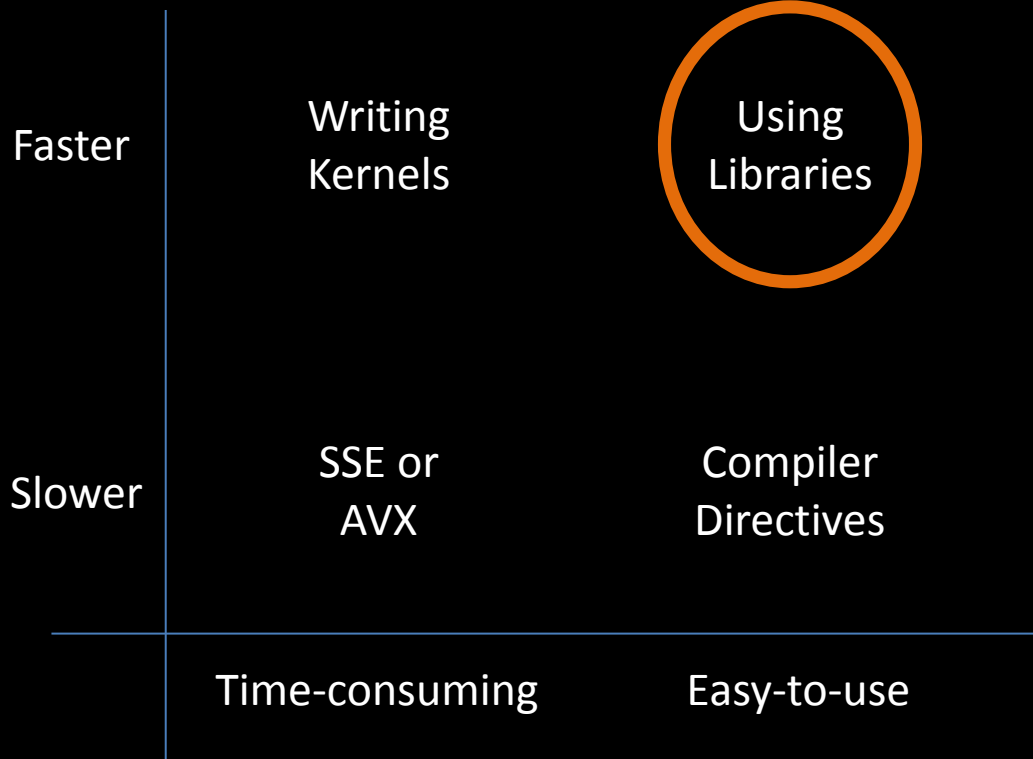
SSE or
AVX

Compiler
Directives

Time-consuming

Easy-to-use

Programmability



Libraries Make the Difference

Raw math libraries in NVIDIA CUDA

- CUBLAS, CUFFT, CULA, Magma
 - Provides *all* BLAS, LAPACK, and FFT routines necessary for most dense matrix operations
- CUSPARSE
 - A good start for sparse linear algebra

Libraries Make the Difference

Raw math libraries in AMD OpenCL

- clAmdBlas, clAmdFft
 - Provides most important Blas routines
 - Provides radix 2, 3, and 5 FFT routines
 - No LAPACK support
 - No sparse data support

Libraries Make the Difference

Raw math libraries in AMD OpenCL

- clAmdBlas, clAmdFft
 - Provides most important Blas routines
 - Provides radix 2, 3, and 5 FFT routines
 - No LAPACK support
 - No sparse data support
 - *Runs on any OpenCL-compliant device*

ArrayFire Code and Benchmarks