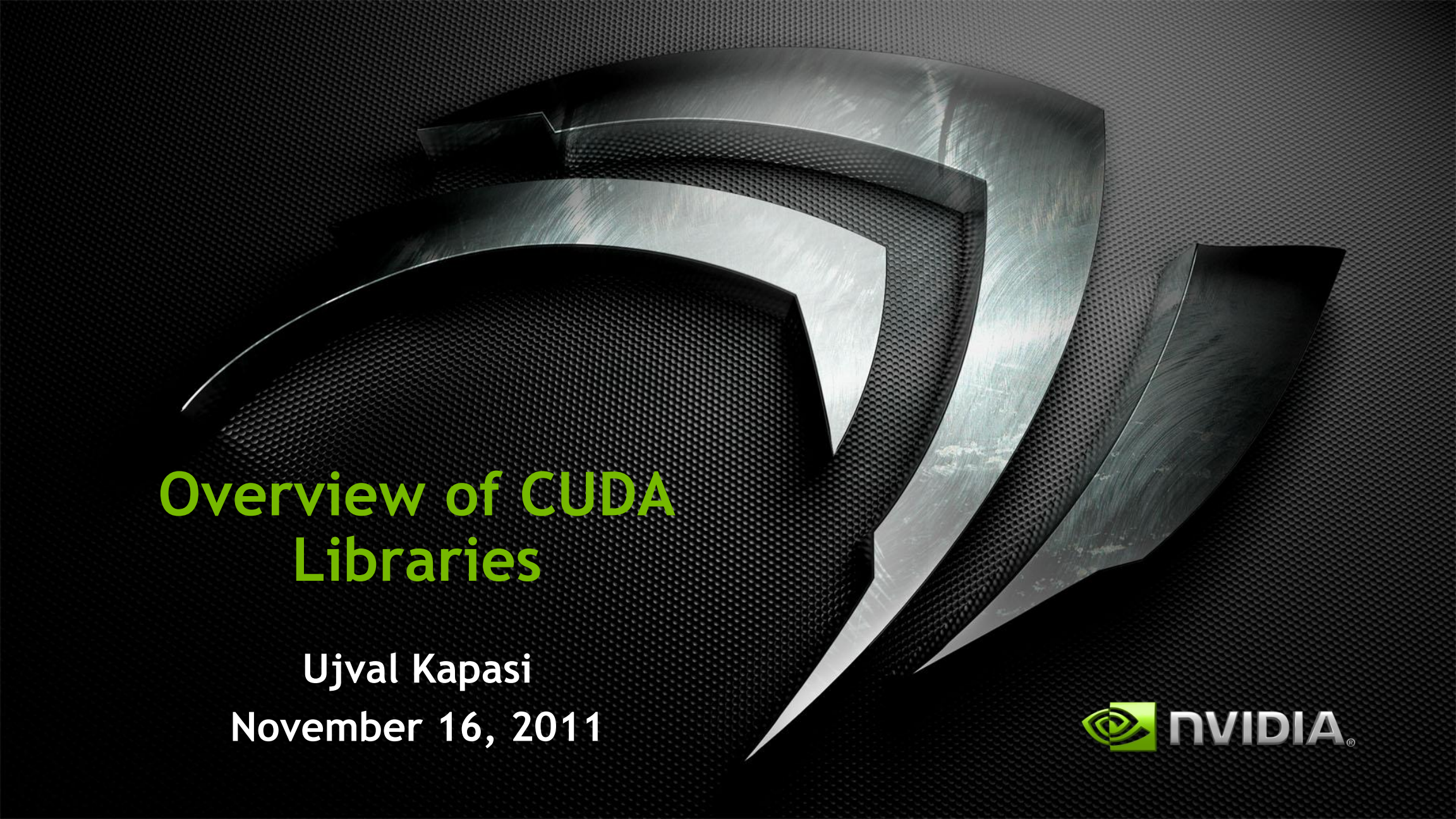
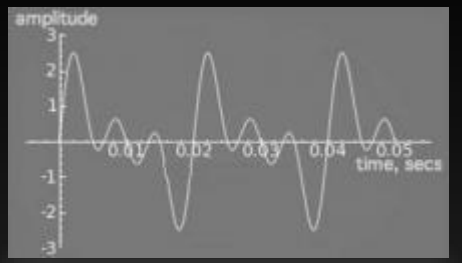

Overview of CUDA Libraries
Ujval Kapasi
November 16, 2011
NVIDIA

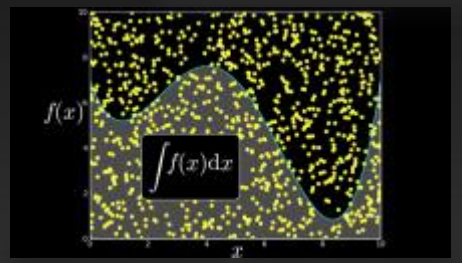
CUDA library ecosystem spans many fields

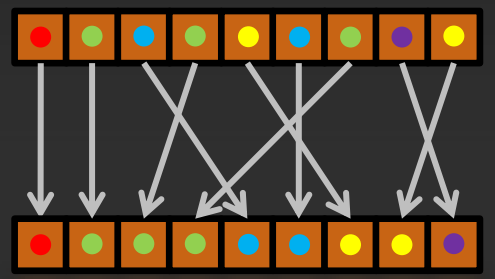
- Math, Numerics, Statistics
- Dense & Sparse Linear Algebra
- Algorithms (sort, etc.)
- Image Processing
- Computer Vision
- Signal Processing
- Finance

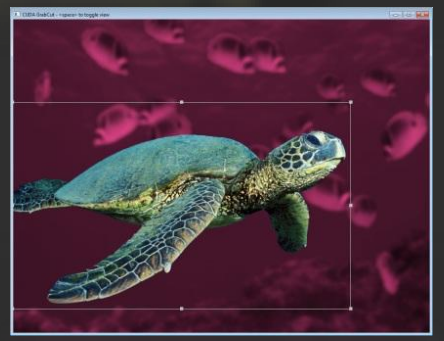
http://developer.nvidia.com/gpu-accelerated-libraries

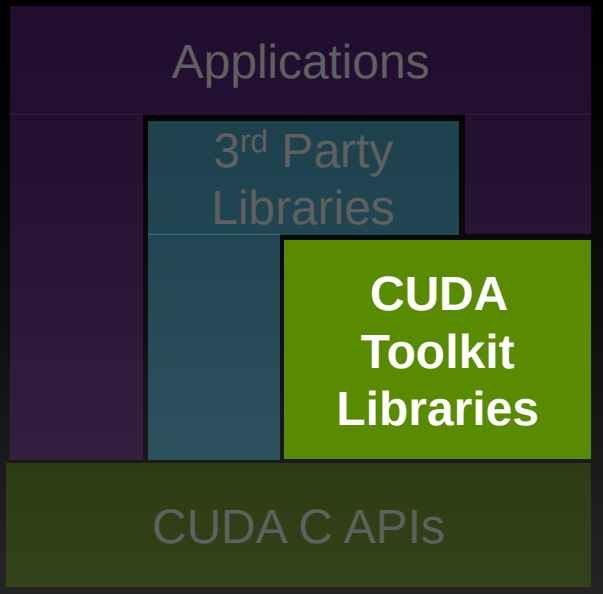
NVIDIA CUDA Toolkit Libraries







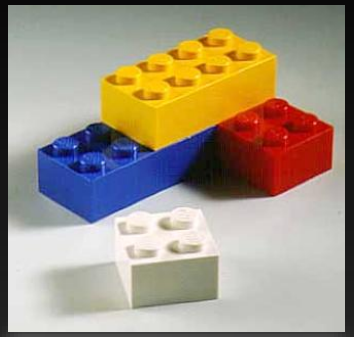




CUBLAS	*dense linear algebra*
CUSPARSE	*sparse linear algebra*
CUFFT	*discrete Fourier transforms*
CURAND	*random number generation*
NPP	*signal and image processing*
Thrust	*scan, sort, reduce, transform*
math.h	*floating point*
system calls	*printf, malloc, assert*

NVIDIA CUDA Library Approach

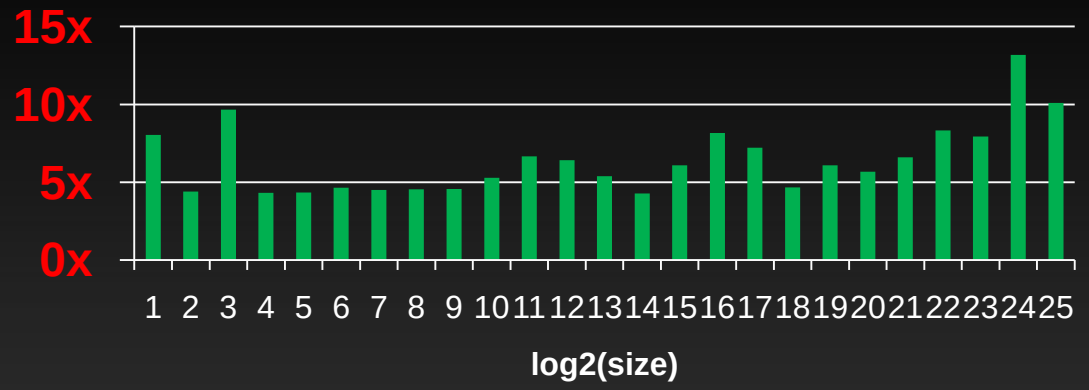
- Provide basic building blocks
- Make them easy to use
- Make them fast



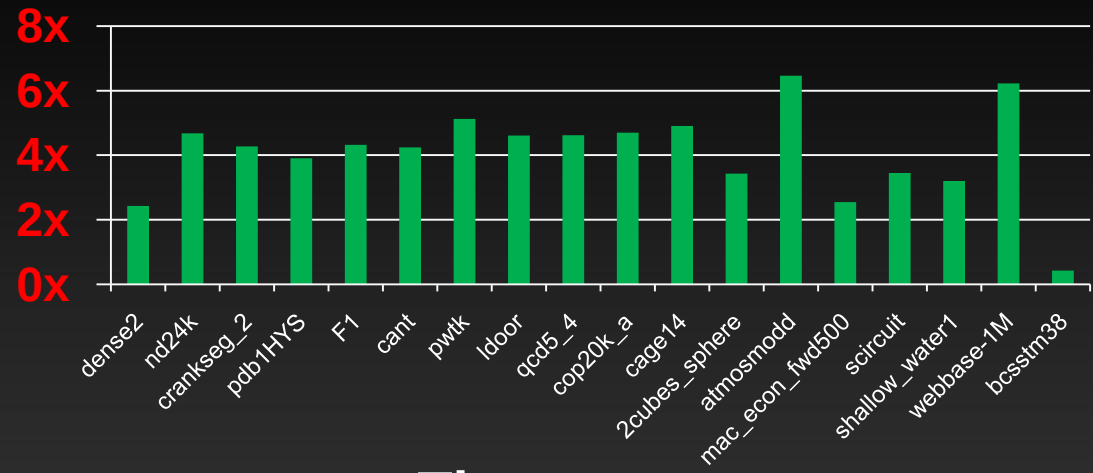
- Provides a quick path to GPU acceleration
- Enables ISVs to focus on their “secret sauce”
- Ideal for applications that use CPU libraries

4x-10x speedups over Intel on single precision

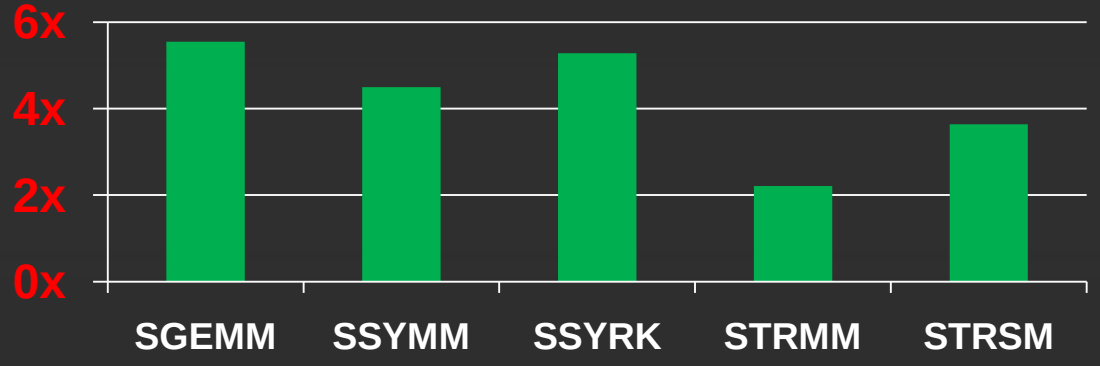
CUFFT



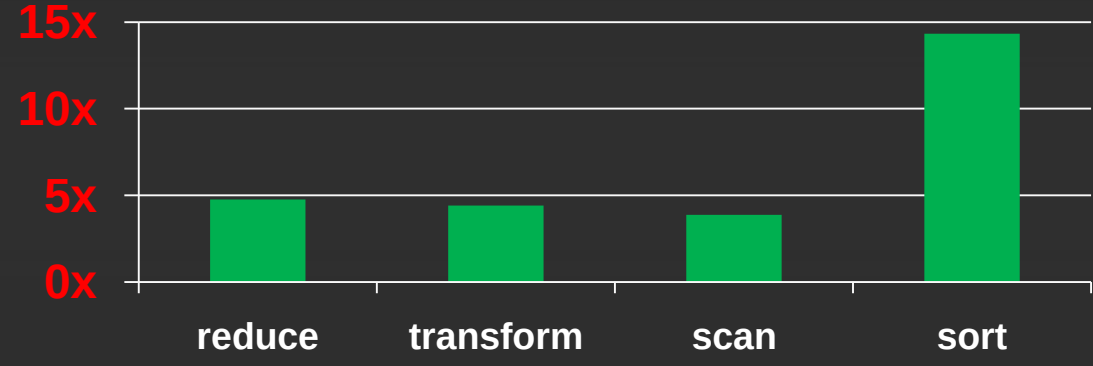
CUSPARSE



CUBLAS

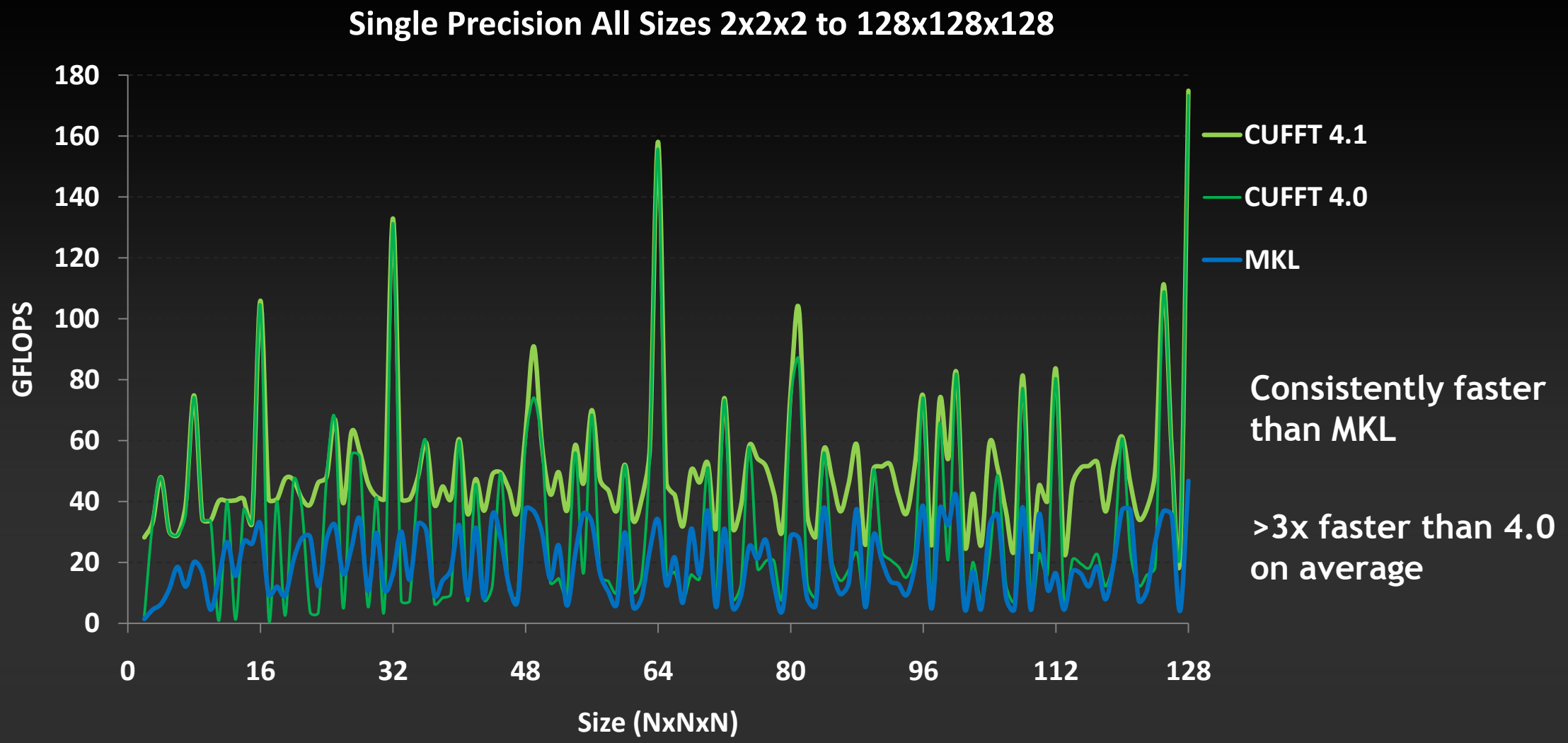


Thrust



- CUDA 4.1 on Tesla M2090, ECC on
- MKL 10.2.3, TYAN FT72-B7015 Xeon x5680 Six-Core @ 3.33 GHz

Constantly improving performance

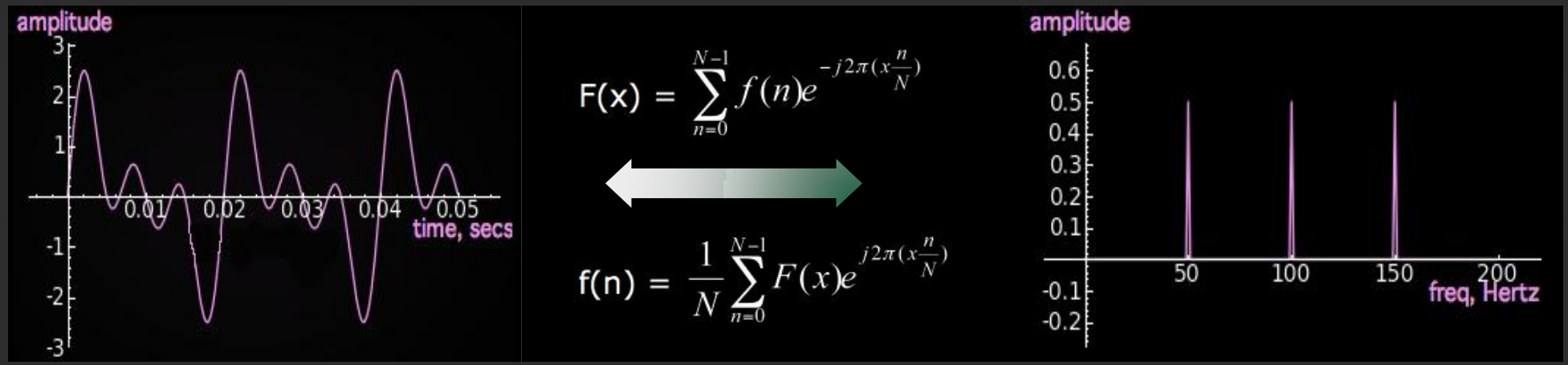


Performance may vary based on OS version and motherboard configuration

- cuFFT 4.1 and cuFFT 4.0 on Tesla M2090, ECC on
- MKL 10.2.3, TYAN FT72-B7015 Xeon x5680 Six-Core @ 3.33 GHz

cuFFT: Multi-dimensional FFTs

- New in CUDA 4.1
  - Flexible input & output data layouts for all transform types
    - Similar to the FFTW "Advanced Interface"
    - Eliminates extra data transposes and copies
  - API is now thread-safe & callable from multiple host threads
  - Restructured documentation to clarify data layouts



cuBLAS: Dense Linear Algebra on GPUs

- Complete BLAS implementation plus useful extensions
  - Supports all 152 standard routines for single, double, complex, and double complex
- New in CUDA 4.1
  - New batched GEMM API provides >4x speedup over MKL
    - Useful for batches of 100+ small matrices from 4x4 to 128x128
  - 5%-10% performance improvement to large GEMMs

cuSPARSE: Sparse linear algebra routines

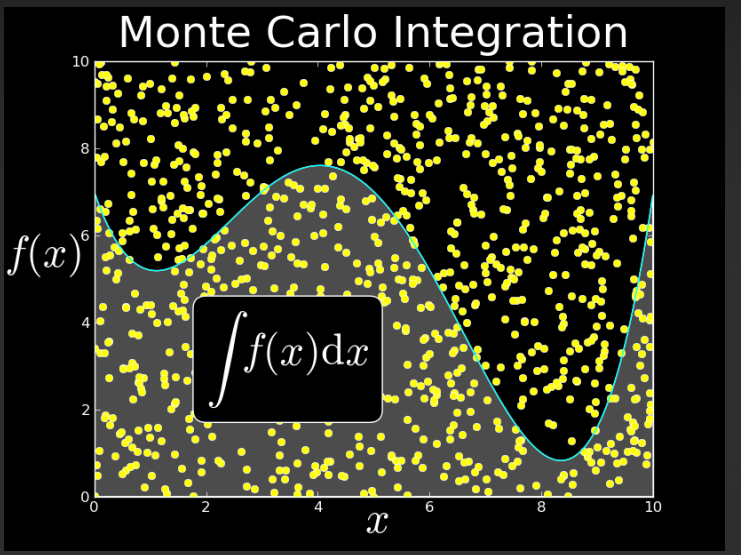
- Sparse matrix-vector multiplication & triangular solve
  - APIs optimized for iterative methods
- New in 4.1
  - Tri-diagonal solver with speedups up to 10x over Intel MKL
  - ELL-HYB format offers 2x faster matrix-vector multiplication

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \alpha \begin{bmatrix} 1.0 & & & \\ 2.0 & 3.0 & & \\ & & 4.0 & \\ 5.0 & & 6.0 & 7.0 \end{bmatrix} \begin{bmatrix} 1.0 \\ 2.0 \\ 3.0 \\ 4.0 \end{bmatrix} + \beta \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

cuRAND: Random Number Generation

- Pseudo- and Quasi-RNGs
- Supports several output distributions
- Statistical test results reported in documentation

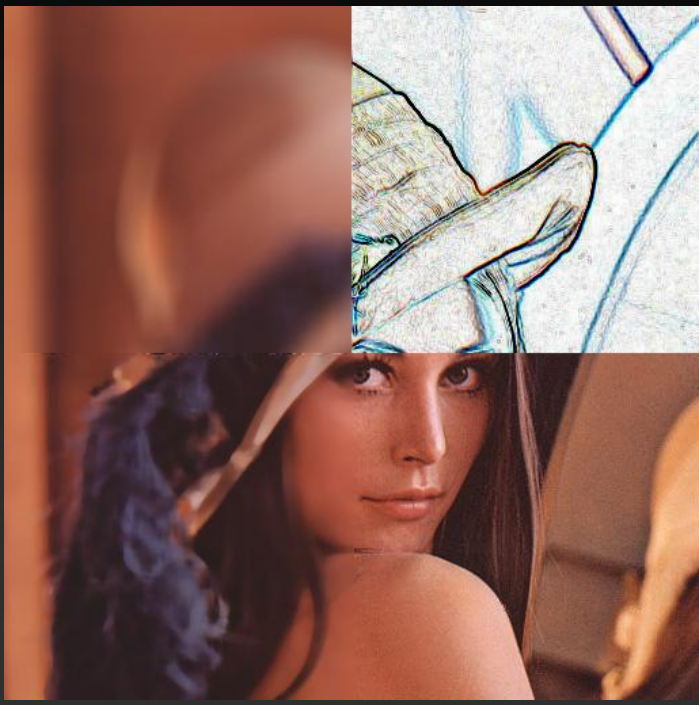
- New commonly used RNGs in CUDA 4.1
  - MRG32k3a RNG
  - MTGP11213 Mersenne Twister RNG



NVIDIA Performance Primitives

Up to **40x** speedups

- Arithmetic, Logic, Conversions, Filters, Statistics, etc.
- Majority of primitives 5x to 10x faster than analogous routines in Intel IPP
- 1,000+ new image primitives in 4.1



* NPP 4.1, NVIDIA C2050 (Fermi)
* IPP 6.1, Dual Socket Core™ i7 920 @ 2.67GHz

Thrust: CUDA C++ Template Library

- Template library for CUDA mimics the C++ STL
  - Optimized algorithms for sort, reduce, scan, etc.
  - OpenMP backend for portability
- Allows applications and prototypes to be built *quickly*
- New in 4.1: Boost-style placeholders allow inline functors
  - Example: *saxpy* in 1 line:

```
thrust::transform(x.begin(), x.end(), y.begin(), y.begin(), a * _1 + _2);
```

math.h: C99 floating-point library + extras

- Basic: +, *, /, 1/, sqrt, FMA (all IEEE-754 accurate for float, double, all rounding modes)
- Exponentials: exp, exp2, log, log2, log10, ...
- Trigonometry: sin, cos, tan, asin, acos, atan2, sinh, cosh, asinh, acosh, ...
- Special functions: lgamma, tgamma, erf, erfc
- Utility: fmod, remquo, modf, trunc, round, ceil, floor, fabs, ...
- Extras: rsqrt, rcbrt, exp10, sinpi, sincos, cospi, erfinv, erfcinv, ...

- **New in 4.1**
  - **Bessel functions: j0, j1, jn, y0, y1, yn**
  - **Scaled complementary error function: erfcx**
  - **Average and rounded average: __{u}hadd, __{u}rhadd**

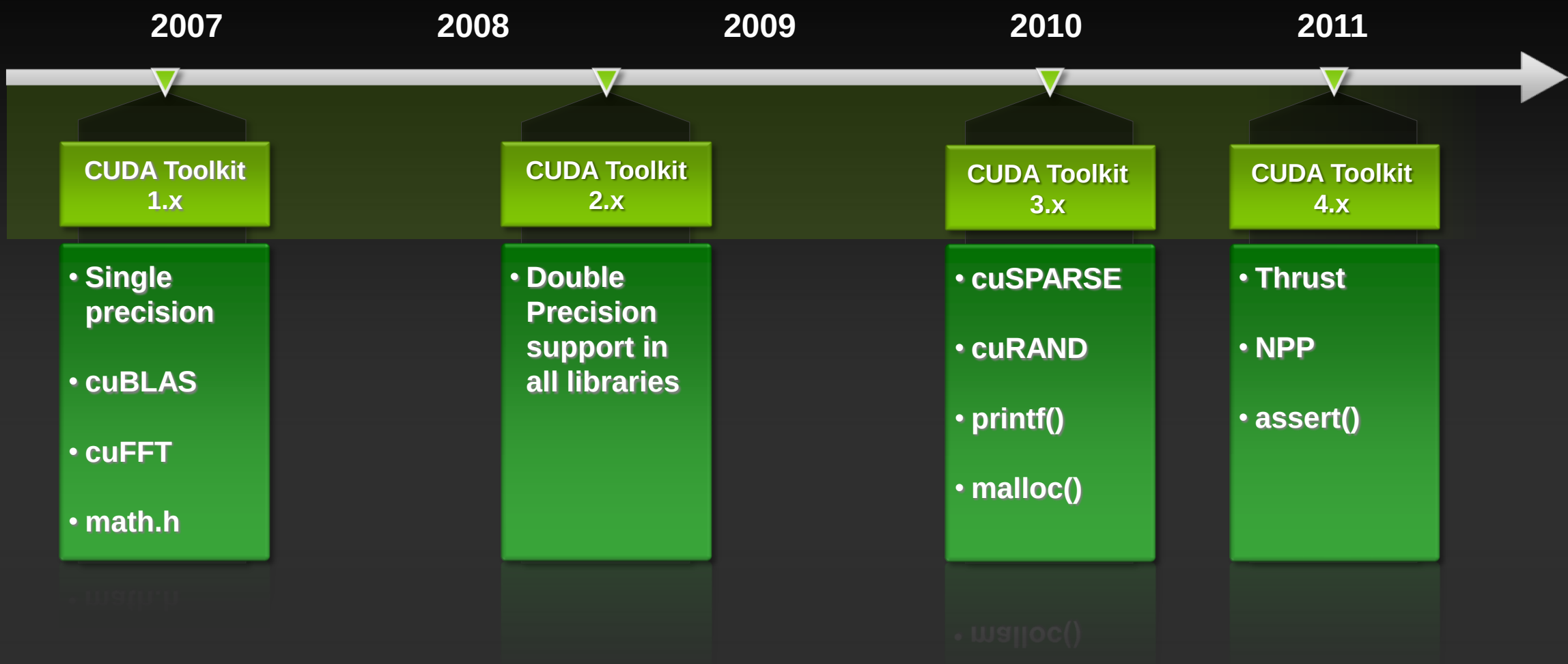
Get started with CUDA libraries today

- CUDA Libraries offer broad coverage of algorithms
  - NVIDIA and 3rd party
- CUDA Library APIs are easy to use
  - Often modeled after widely used APIs for CPU libraries
  - I.e., CUBLAS → BLAS, CUFFT → FFTW, Thrust → STL
- CUDA Libraries are heavily optimized for NVIDIA GPUs
  - Automatic performance improvements with new CUDA releases and new GPU architectures


Thank You!
NVIDIA

Backup

Constant progress on library development



Overall library ecosystem

- Linear Algebra
  - CUBLAS, CUSPARSE, CUSP, MAGMA, CULA, CULA Sparse phiGEMM, libFLAME, FMSLib
- Math / Numerics
  - CURAND, NAG, libjacket, IMSL
- Algorithms
  - Thrust, CUDPP
- Image Processing and Computer Vision
  - NPP, OpenVIDIA, OpenCV, cuvi, VSIPL
- Signal Processing
  - CUFFT , NukadaFFT
- Finance
  - Kooderive

New functionality in 4.1

- Over 1,000 image processing functions added to NPP
- Sparse tri-diagonal solver added to CUSPARSE
- Widely used random number generators added to CURAND
  - MRG32k3a and Mersenne Twister (MTGP, based on MT11213)
- Additions to math.h
  - Bessel functions: j0, j1, jn, y0, y1, yn
  - Scaled complementary error function: erfcx
  - Average and rounded average: __{u}hadd, __{u}rhadd

Productivity improvements in 4.1

- Boost-style placeholders added to Thrust
  - These allow functors to be concisely defined inline
  - Example: *saxpy* in 1 line:

```
thrust::transform(x.begin(), x.end(), y.begin(), y.begin(), a * _1 + _2);
```

Performance improvements in 4.1 (for Fermi)

- "Batched" GEMM API in CUBLAS performs many small matrix multiplies
- Sparse matrix-vector multiply in CUSPARSE is up to 2x faster using ELL/HYB format
- Significant optimization to IEEE floating-point operations implemented in software (i.e., divide, reciprocal, and square-root)
- Many other improvements are documented in the release notes