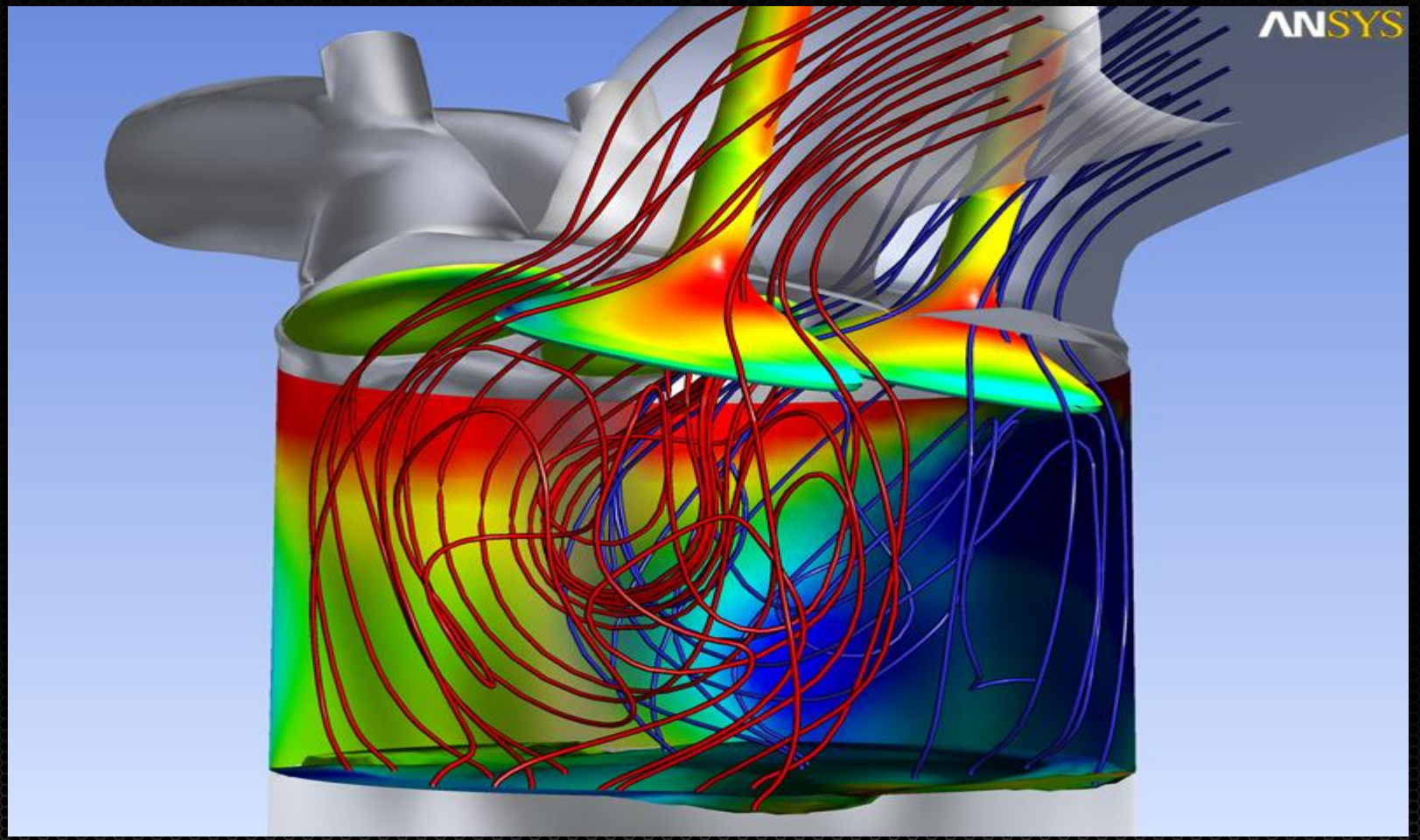
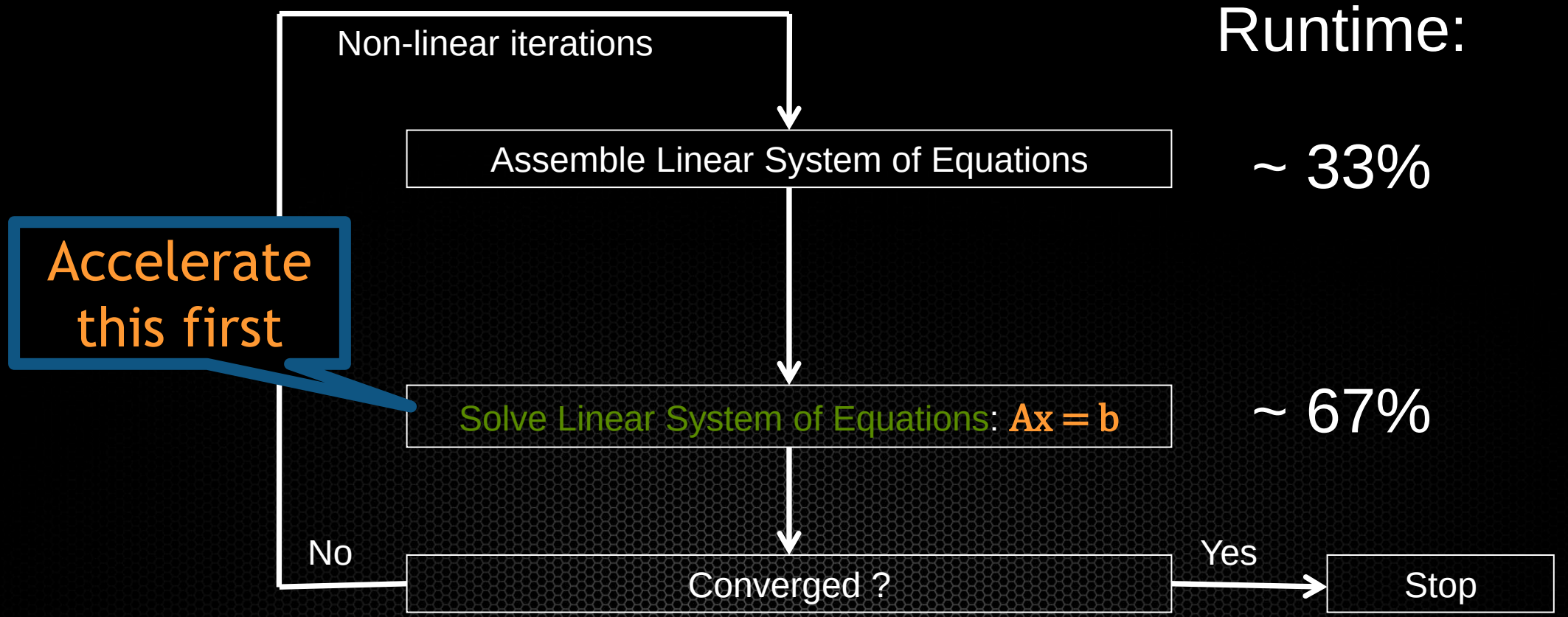
NVIDIA MPI-enabled Iterative Solvers for Large Scale Problems

Joe Eaton
Manager, AmgX CUDA Library
NVIDIA

ANSYS Fluent



Fluent control flow



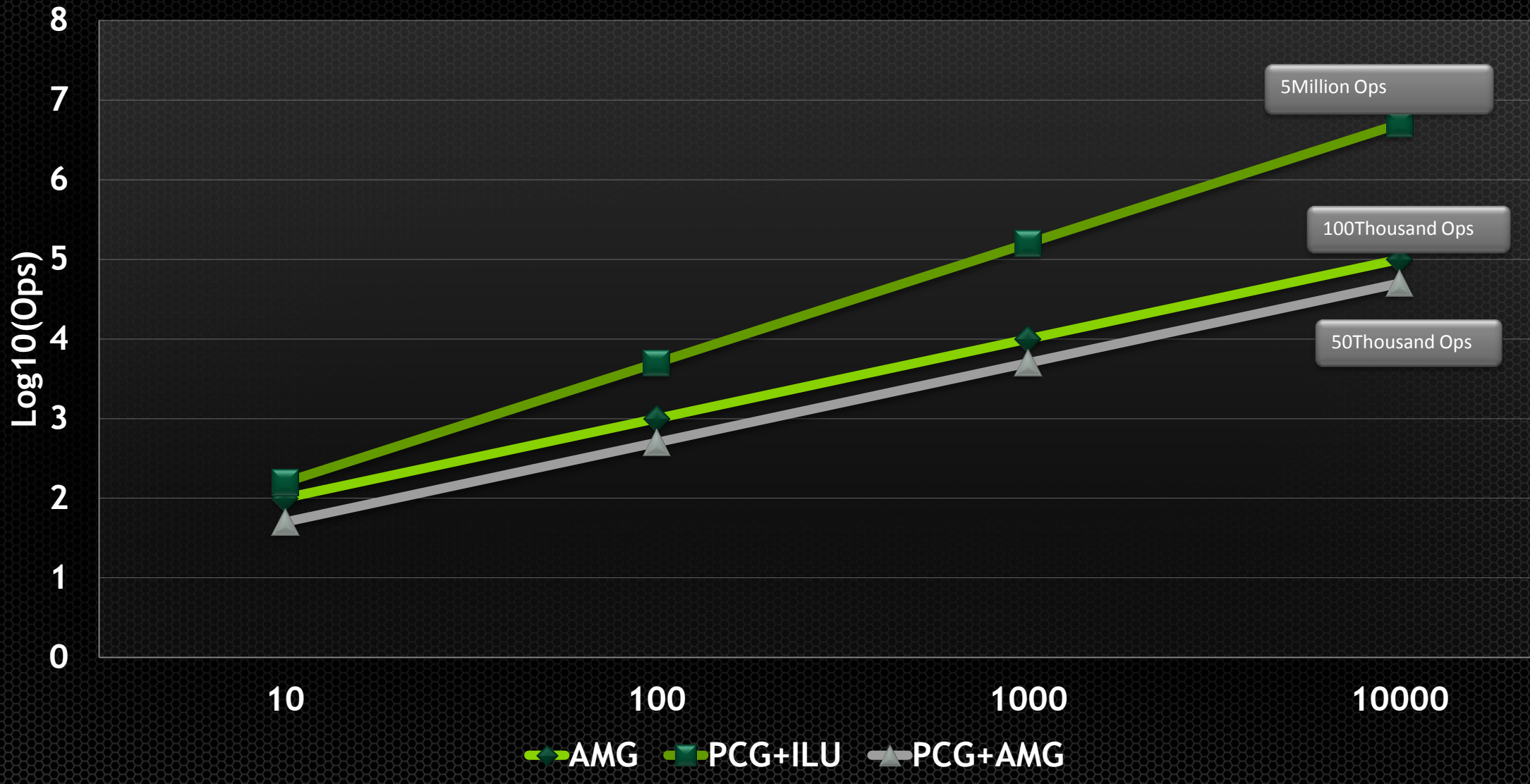
Break it Down

Iterative Solvers

- Low memory requirements
- Best choice if only a few digits of accuracy are needed
- Optimal complexity: O(N)

- Multigrid methods
  - Optimal for discretized elliptic and mixed elliptic-hyperbolic type PDEs
  - For a black-box method, choose Algebraic Multigrid (AMG)
- Krylov methods
  - PCG, GMRES, BiCGSTAB, IDR-> these work together with AMG, complementary
  - Easy to create parallel implementations

Scalability- LogLog of Ops vs N



Multigrid on GPUs

Notice I didn't claim Multigrid was easy to implement in parallel ☺

- For GPUs, we need fine-grain parallelism, in addition to DD
- MPI for coarse grain parallelism, CUDA + OpenMP on each node.
- NVIDIA set out to create a world-class AMG implementation on GPUs

3 years of effort, 7 Ph.D. contributors, 9 Patents later

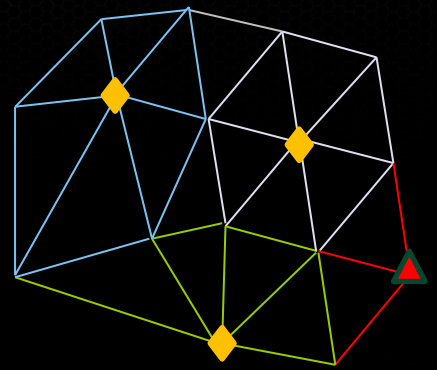
AmgX is HERE!

Parallel AMG (Classical and Aggregation)

- Smoothers (Block-Jacobi, GS, DILU, Polynomial)
- MG cycles (V,W,F,CG)

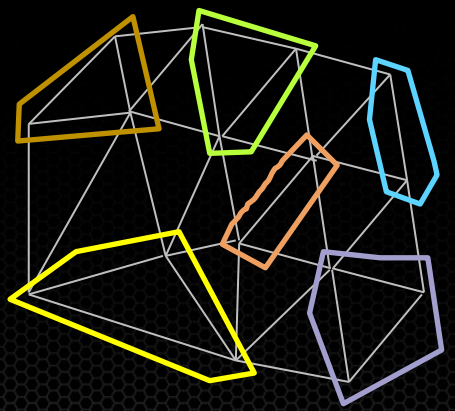
Classical Ruge-Steuben AMG

- ✓ **Robust convergence**
- \- High setup cost, two phases
- • Only scalar matrices



Unsmoothed Aggregation AMG

- ✓ Easier to parallelize
- ✓ Lower grid and Operator complexity
- ✓ Block structured matrices



Extras

- Preconditioned & Accelerated Eigensolvers
  - Power method, inverse power method, shifted inverse
  - LOBPCG
  - Jacobi-Davidson
  - Subspace Iteration

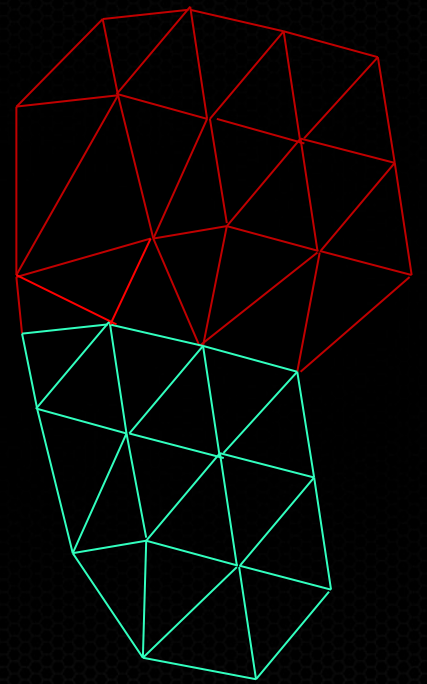
- $Ax=\lambda x$
- Find some extremal eigenvalues fast
- Find eigenvalues near a given value

Krylov Methods

- All Rely on massively parallel SpMV
  - CG,PCG, PCGF
  - GMRES, FGMRES, DQGMRES
  - BiCGStab, BiPCGStab
  - IDR(s)
- IDR(s) is the new kid on the block, allows a smooth variation between BiCGStab behavior and GMRES-like behavior

Graph Utilities

- Partitioning with METIS
- Graph coloring for parallel smoothers

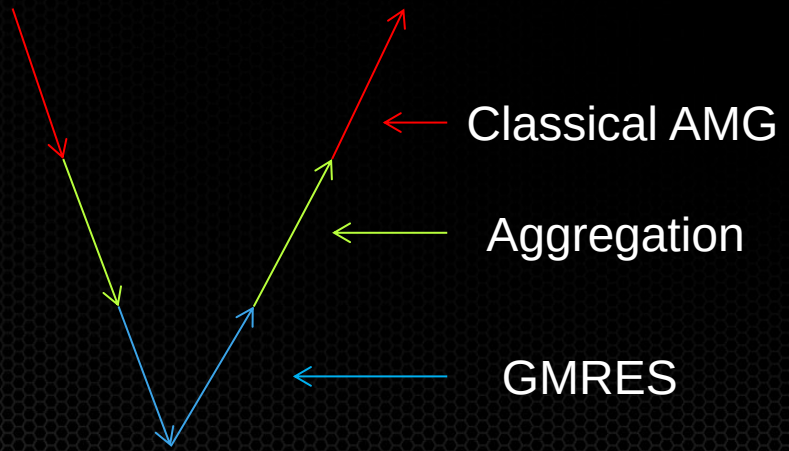


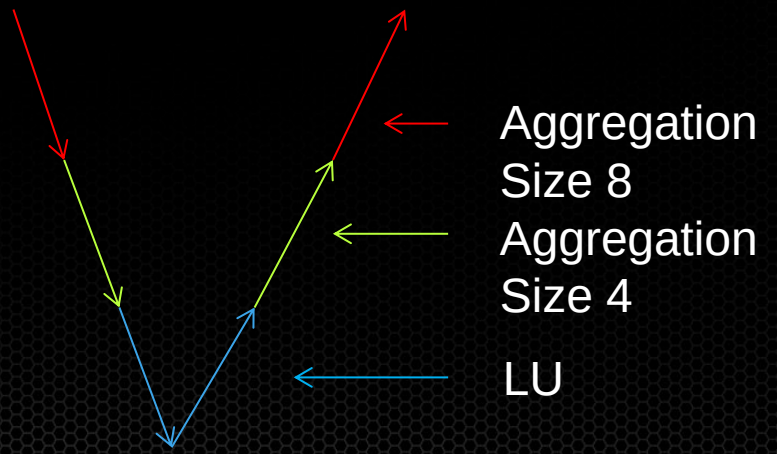
- Produce fast and high-quality results

New non-zero-counting load balancing

Flexible Solver Construction

- All methods composable, nestable to create complex solvers easily (PCG+AMG+GMRES, GMRES+AMG, AMG+AMG)
- Offers unique NVIDIA methods for massive parallelism





Standards Compliance

- Supports standard matrix formats
  - Matrix-Market, CSR, BSR, COO
- Easy-to-integrate C API
- Tested with multiple MPI implementations
  - PCMPI, MVAPICH, OpenMPI
- Works with CUDA 5.0 and CUDA 5.5

Easy to Use

- No CUDA experience necessary to use the library
- Small API (<30 calls), no bloat
- Upload a matrix from the host, or hand it over in device memory
- Uses CUDA Unified Virtual Addresses for zero-copy uploads
- Supports NVIDIA GPUs Fermi and newer
- Single, Double, Mixed precision
- Handles consolidation of matrix on fine or coarse level
- Single GPU, Multi-GPU, clusters with MPI

Minimal Example- 11 calls

```
AMGX_initialize();
AMGX_initialize_plugins();
AMGX_create_config(&cfg, cfgfile);
AMGX_resources_create_simple(&res, cfg);
AMGX_solver_create(&solver, res, mode, cfg);
AMGX_matrix_create(&A,res,mode);
AMGX_vector_create(&x,res,mode);
AMGX_vector_create(&b,res,mode);
AMGX_read_system(&A,&x,&b, matrixfile);
AMGX_solver_setup(solver,A);
AMGX_solver_solve(solver, b, x);
```

Capacity & Performance

- We have verified 3 million rows per GB of GPU memory

-> 12 million rows on Tesla K20x (6GB)

-> 24 million rows on Tesla K40 (12GB)

- Single GPU performance typically 5million rows /sec for 1 digit

-> 10 million rows solved to 6 digits (1e-6 relative tolerance)

10/5 * 6= 12 seconds

-> 8 million row system, to 10 digits

8/5 * 10 = 16 seconds

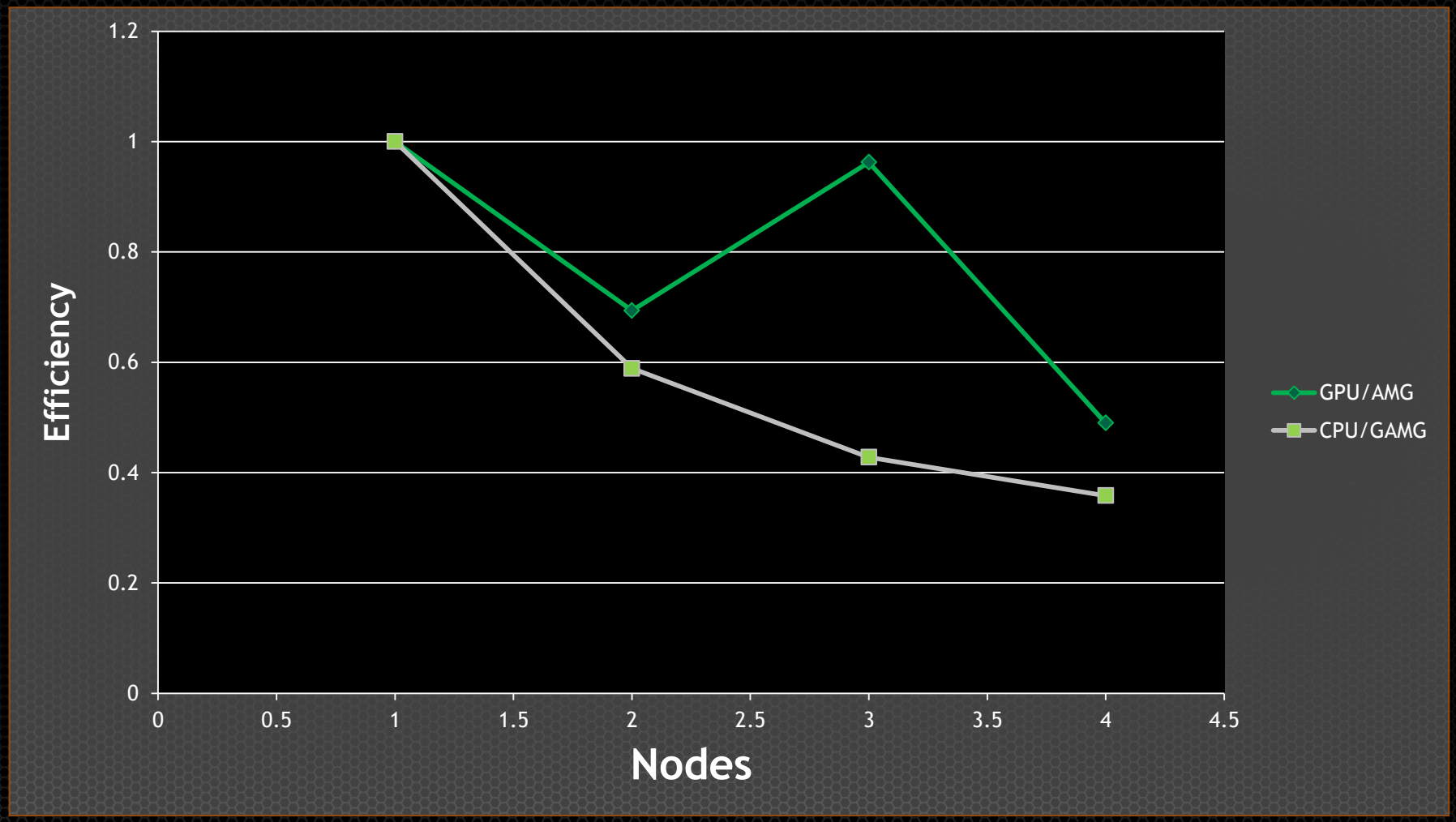
Your mileage may vary: miniFE app delivers 7.44 M row/sec for 1 digit

MPI Performance

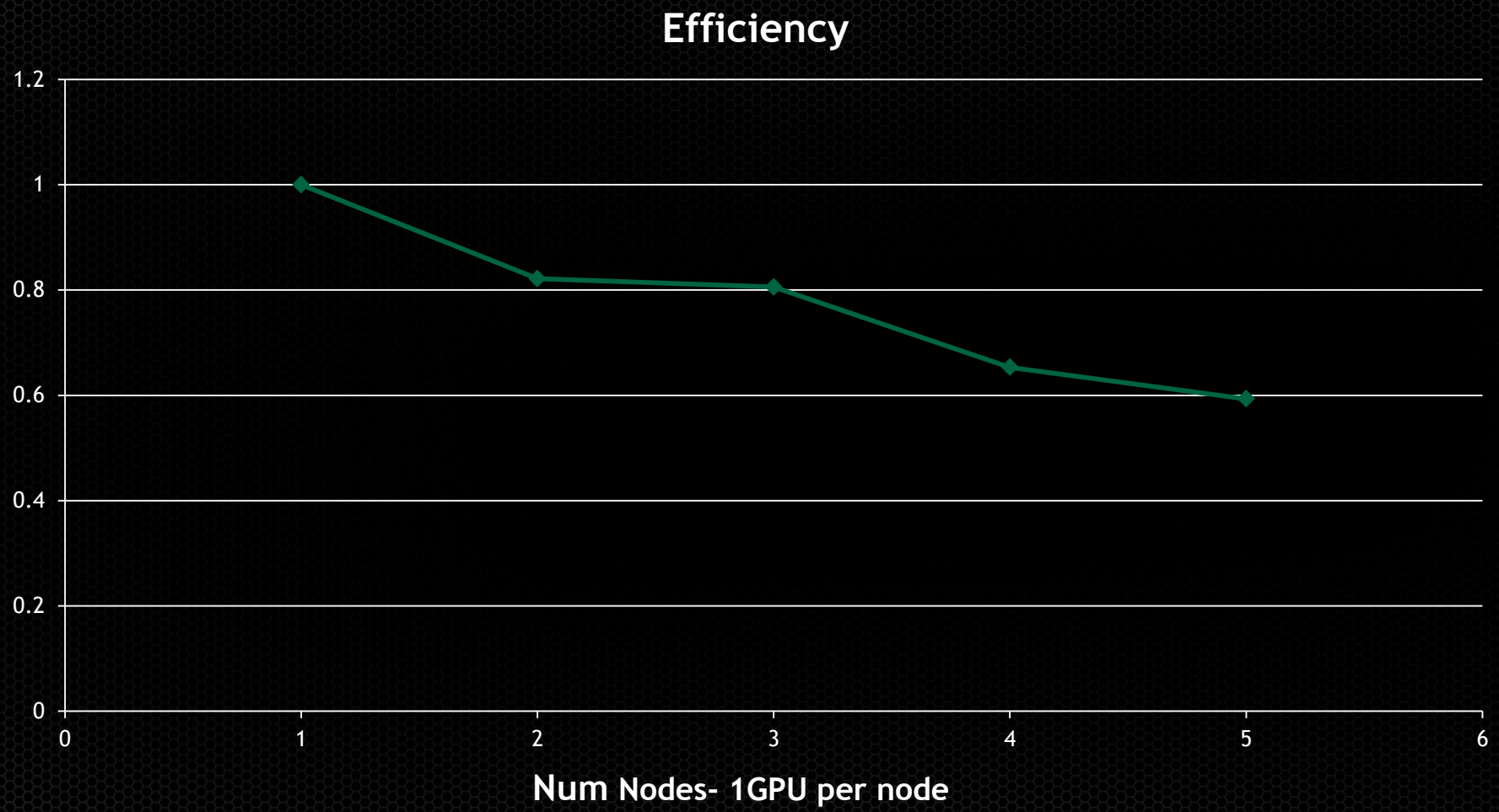
- Dominated by coarse grid communication overhead
  - ✓ Consolidate coarse grids to a single GPU
  - ➢ Try out asynchronous coarse corrections

  - When we engage multiple nodes, we see big hit
  - Weak scaling good to 4 GPUs (95% efficient)
  - Above 16 GPUs, we have 50% scaling for

OpenFOAM Weak Scaling



Strong Scaling Fluent



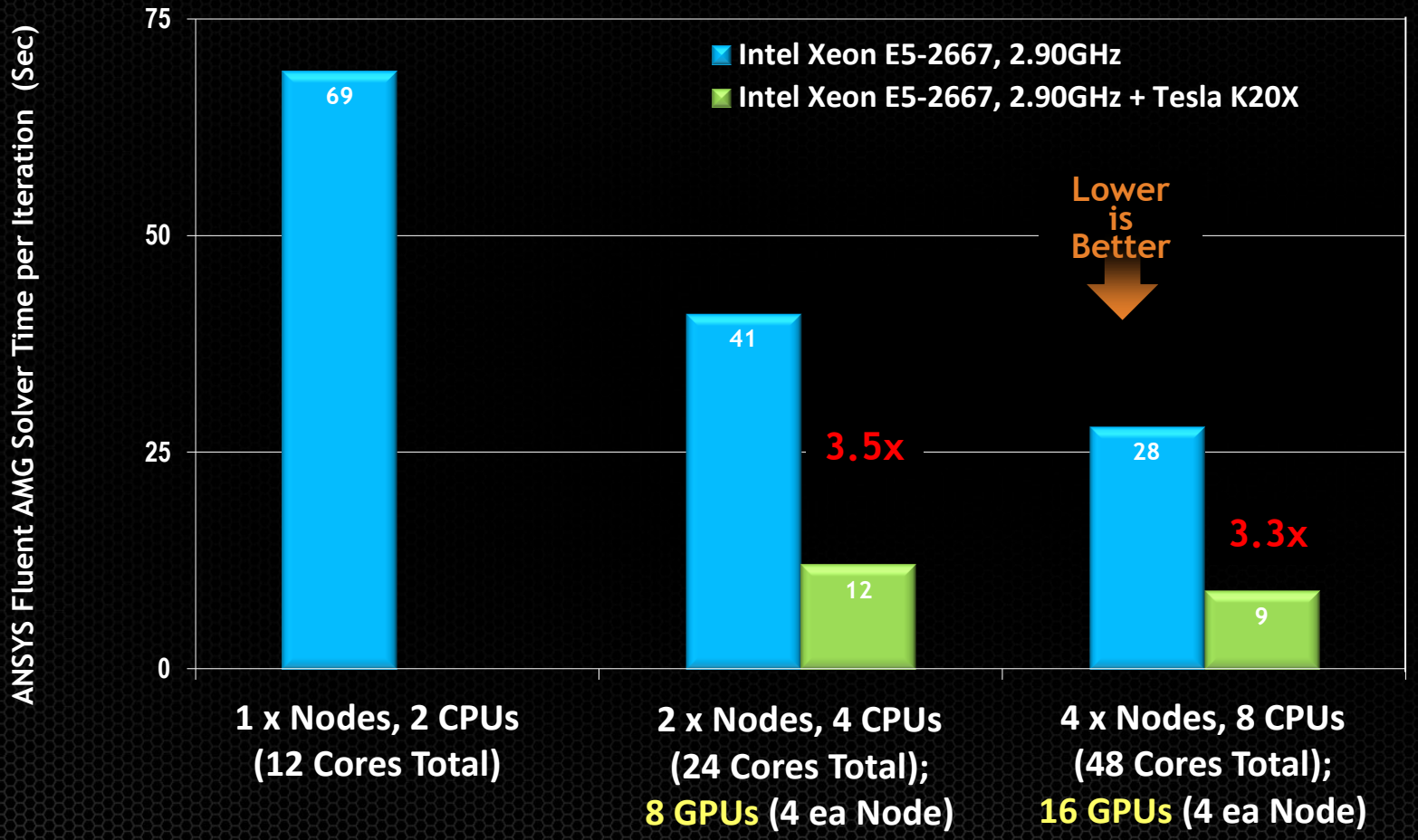
Strong Scaling 3M rows, Single Node

Beta Users

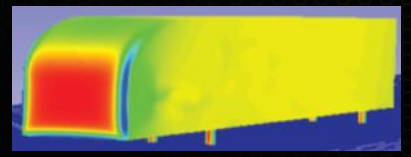
- We have started a private Beta test program,14 institutions and ISVs
- Public release in Nov, 2013 at SC'13
- Use cases so far:
  - Oil&Gas (exploration and production)
  - CFD (external aerodynamics,incompressible and compressible flow)
  - Flooding and storm surge simulation
  - Tsunami simulation
  - Coupled heat transfer and flow in pipes
  - Network/graph metrics (page rank, maximal flow, partitioning)
  - Computer Vision

ANSYS Fluent for 14M Cell Aerodynamic Case

ANSYS Fluent 15.0 Preview Performance - Results by NVIDIA, Jun 2013



Truck Body Model

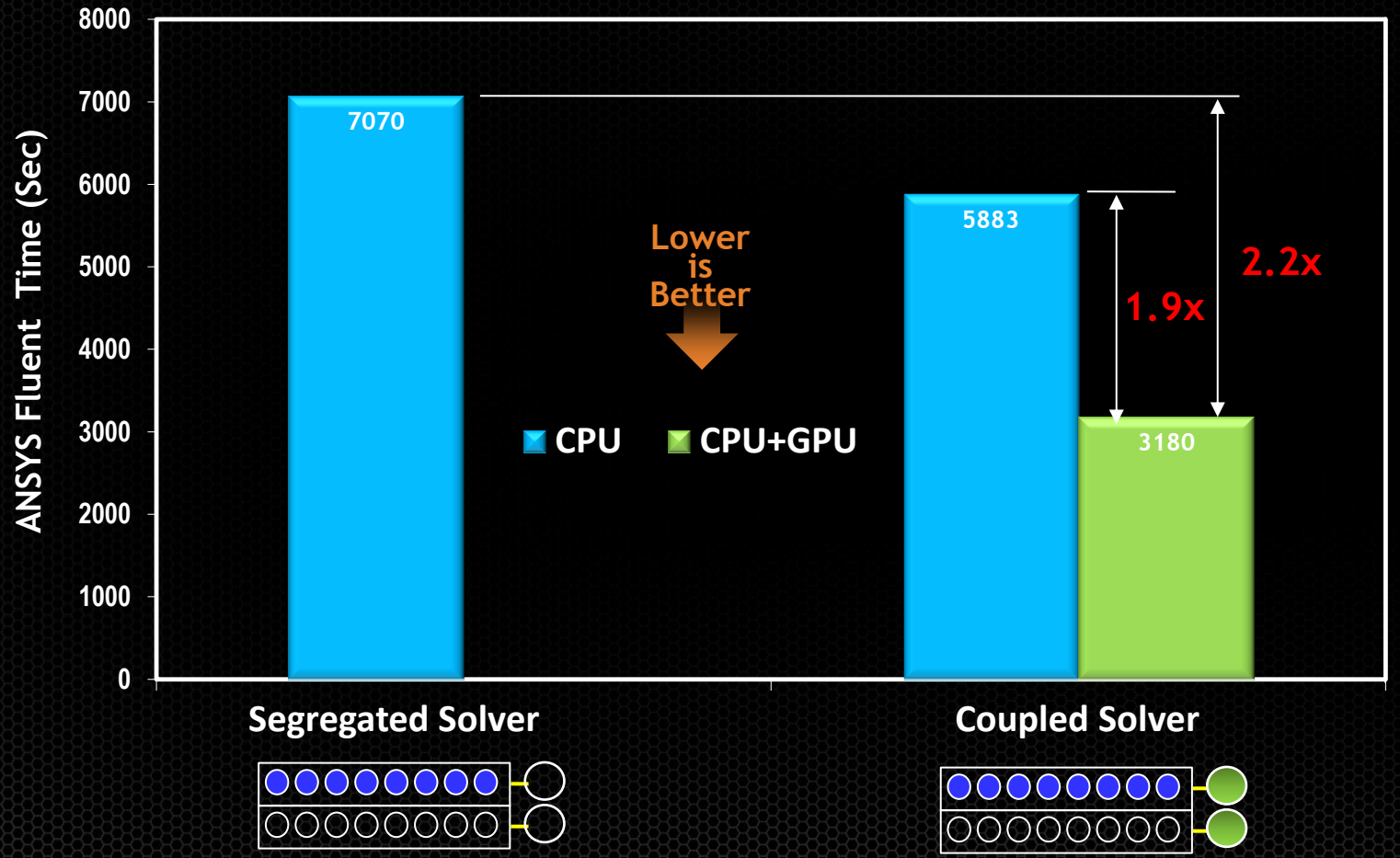


- 14 M Mixed cells
- DES Turbulence
- Coupled PBNS, SP
- Times for 1 Iteration
- AMG F-cycle on CPU
- GPU: Preconditioned FGMRES with AMG

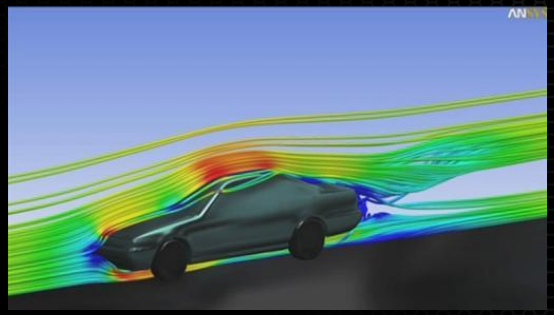
NOTE: All jobs solver time only

GPU Acceleration on Coupled Solvers

Preview of ANSYS Fluent 15.0 Performance



Sedan Model



- Sedan geometry
- 3.6 M mixed cells
- Steady, turbulent
- External aerodynamics
- Coupled PBNS, DP
- AMG F-cycle on CPU
- AMG V-cycle on GPU

NOTE: Times for total solution

Tremendous Collaboration and Team

- Thanks to an awesome team (in alphabetical order):

  Marat Arsaev, Patrice Castonguay, Jonathan Cohen, Julien Demouth, Bhushan Desam, Joe Eaton, Justin Luitjens, Nikolay Markovskiy, Maxim Naumov, Stan Posey, Nikolai Sakharnykh, Robert Strzodka, Han Van Holder, Zhenhai Zhu

- Star interns:

  Peter Zaspel, Simon Layton, Lu Wang, Istvan Reguly, Francesco Rossi, Christoph Franke, Felix Abecassis, Rohit Gupta

- Our collaborators and AMG advisors:

  ANSYS: Sunil Sathe, Prasad Alavilli, Rongguang Jia
  PSU: James Brannick, Ludmil Zikatanov, Jinchao Xu, Xiaozhe Hu Developers at companies to be named later…