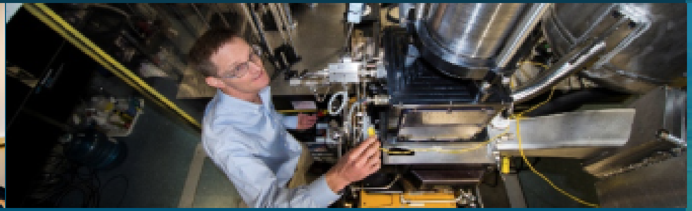
SAND2022-2126C

Sandia National Laboratories

Preparing Trilinos Solvers for Exascale Wind Farm Simulations



PRESENTED BY Jonathan Hu, Luc Berger-Vergiat, Ichitaro Yamazaki

SIAM Parallel Processing

Wednesday, February 23, 2022

U.S. DEPARTMENT OF ENERGY NNSA

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline

ExaWind project overview

Role of linear solvers

Multigrid and Trilinos

Numerical Results

Ongoing and Future Work

Exawind Project Overview

ExaWind Goals

- Create a multi-fidelity modeling and simulation environment for wind turbines and wind farms
- Enable simulations on current and next-generation supercomputers
- Enable a new understanding and ability to predict wind farm flows and turbine responses
- Create a foundation for next-generation lower-fidelity engineering models

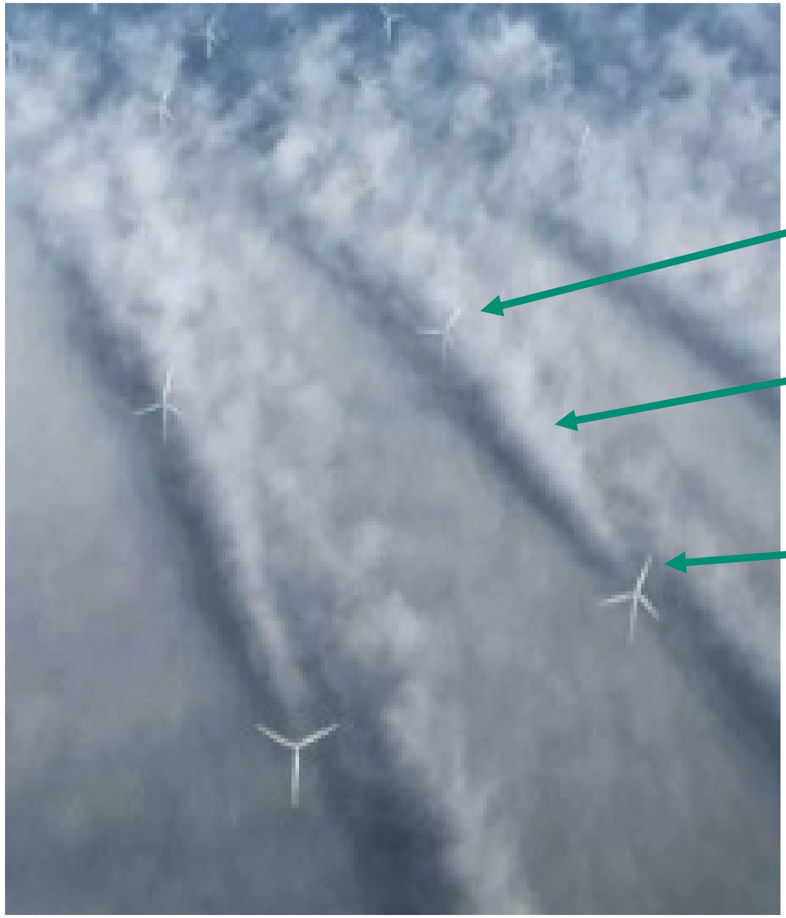


Photo by Gitte Nyhus Lundorff, Bel Air Aviation Denmark – Helicopter Services

Can we predict and understand:

Impact of wakes on downstream turbines?

Evolution of the wakes?

Formation of the wakes?

… and all in a highly complex, dynamic metocean environment

Slide courtesy of M. Sprague (NREL)

ExaWind primary application codes

Nalu-Wind

- https://github.com/exawind/nalu-wind
- Incompressible-flow computational fluid dynamics (CFD) code
- Unstructured-grid finite-volume discretization
- Closely tied to Trilinos
  - Iterative linear-system solvers
  - Algebraic multigrid preconditioners
  - Kokkos abstraction layer
  - STK mesh data structures
- Can also utilize *hypre* solvers & preconditioners
- Critical for blade-resolved simulations

AMR-Wind

- https://github.com/Exawind/amr-wind
- Incompressible-flow CFD code
- Structured-grid finite-volume **background solver with adaptive mesh refinement (AMR)**
- Built on AMReX library
- Multi-level geometric multigrid linear-system solvers
- Coupled to Nalu-Wind through overset meshes
- Can utilize *hypre* solvers & preconditioners

TIOGA

- https://github.com/jsitaraman/tioga
- Overset mesh coupling

Slide adapted from M. Sprague (NREL)

For purposes of this talk, we focus on the linear solvers in Nalu-Wind.

Role of Linear Solvers in Nalu-Wind

Nalu-Wind solves the incompressible Navier Stokes equations

Momentum and continuity phases require solution of large sparse linear systems.

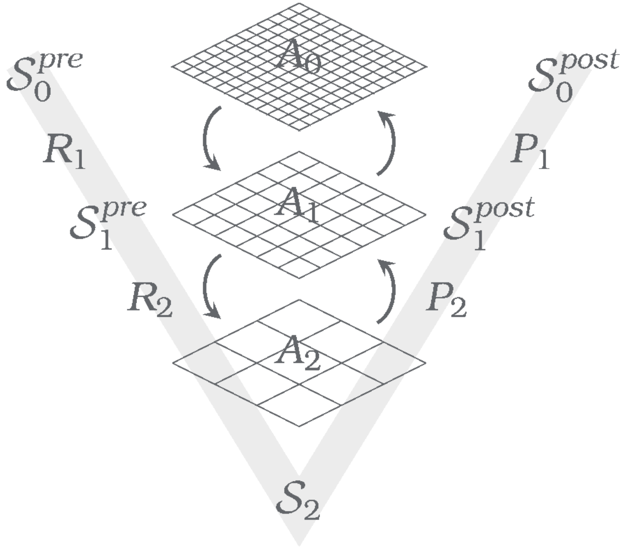
Matrices and thus solvers must be rebuilt for every solve.

Efficient Krylov solvers and scalable preconditioners are necessary.

Multigrid is a natural fit.

Multigrid Introduction

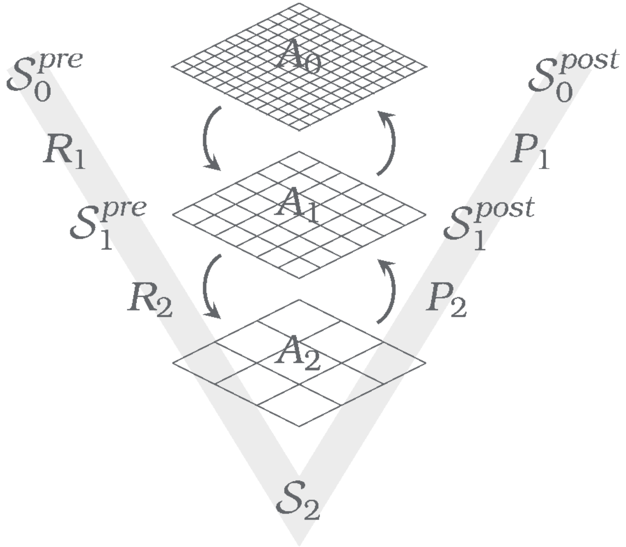
- Scalable solution method for linear systems arising from elliptic PDEs
- Often used as preconditioner to Krylov method
- Idea: capture error at multiple resolutions:
  - **Smoothing** reduces oscillatory error (high energy)
  - **Coarse grid correction** reduces smooth error (low energy)



Multigrid Introduction

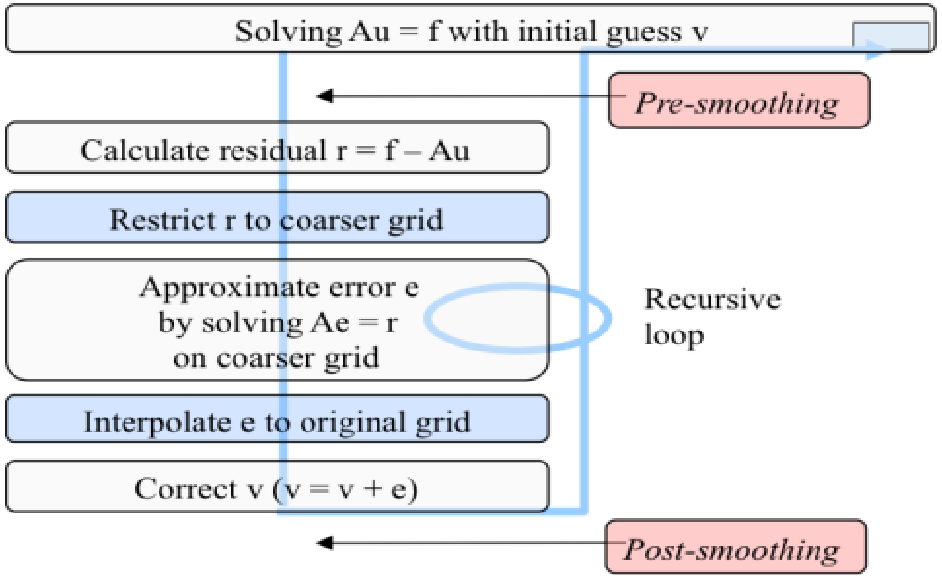
- Scalable solution method for linear systems arising from elliptic PDEs
- Often used as preconditioner to Krylov method
- Idea: capture error at multiple resolutions:
  - **Smoothing** reduces oscillatory error (high energy)
  - **Coarse grid correction** reduces smooth error (low energy)

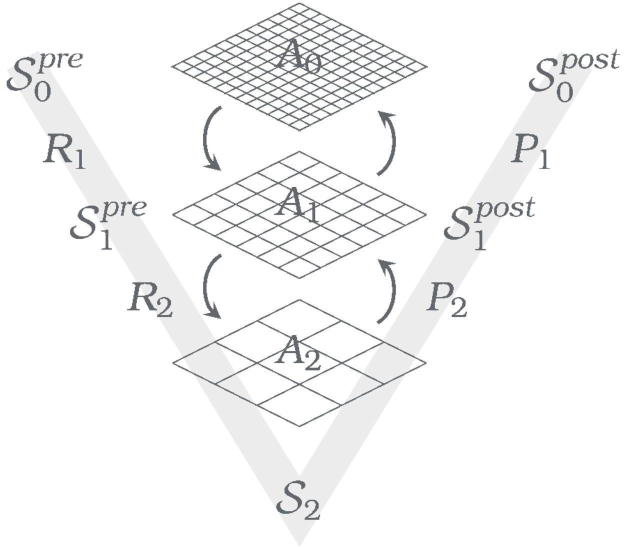
- Geometric multigrid (GMG)
  - Application supplies A_i's, R_i's, and P_i's
- Algebraic multigrid (AMG)
  - Preconditioner generates A_i's, R_i's, P_i's
  - Two ways to coarsen
    - Ruge Stueben (coarse DOFs subset of fine)
    - Aggregation (group fine DOFs to form coarse)



Multigrid Introduction

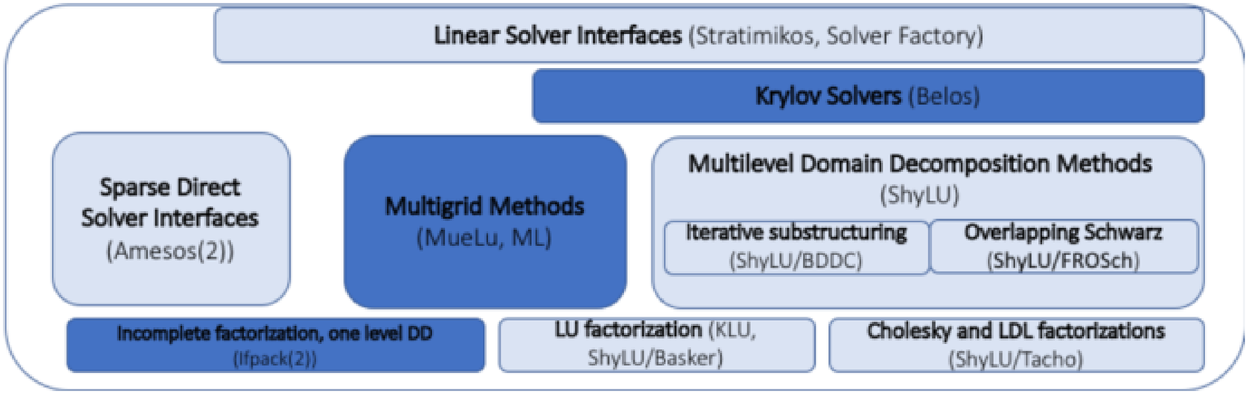
- Scalable solution method for linear systems arising from elliptic PDEs
- Often used as preconditioner to Krylov method
- Idea: capture error at multiple resolutions:
  - **Smoothing** reduces oscillatory error (high energy)
  - **Coarse grid correction** reduces smooth error (low energy)





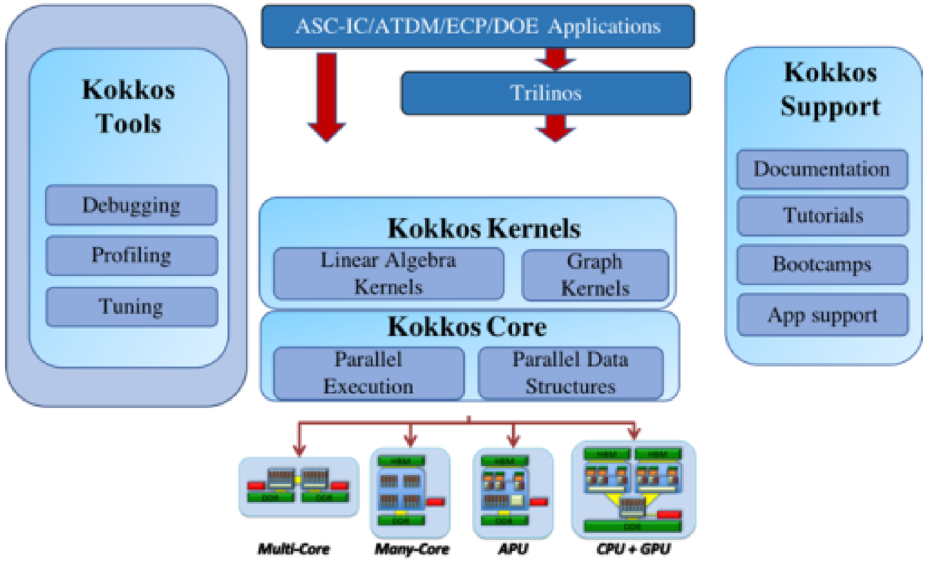
Software

Trilinos Project



github.com/trilinos/Trilinos

github.com/kokkos



MueLu Multigrid Library

Unstructured algorithms

- classic smoothed aggregation (SA)
- non-symmetric AMG
- AMG for Maxwell's equations

Structured Algorithms

- semi-coarsening AMG
- geometric MG
- structured-grid aggregation-based MG

Leverages many other Trilinos scientific libraries

- Shared memory parallelism from Kokkos → architecture portability
- Sparse distributed linear algebra: Tpetra
- Distributed smoothers: Ifpack2
- Shared memory smoothers, SpGEMM, distance-2 coloring: Kokkos-Kernels
- Load balancing: Zoltan2
- Direct Solvers: Amesos2

Numerical Results

Numerical Experiments

Rotating wind turbine simulation using refined version of NREL5MW mesh

- 5MW reference wind turbine (Jonkman et al., NREL Tech Report #TP-500-38060, 2009)
- 634.5e6 nodes, 719.4e6 elements
- “hybrid” mesh: extruded structured around blade, unstructured around hub and hub/blade transition

Experiments run on ORNL Summit supercomputer

- 4600 compute nodes, each with two Power9 CPUs, six NVIDIA Tesla V100 GPUs

ExaWind is primarily interested in strong-scaling

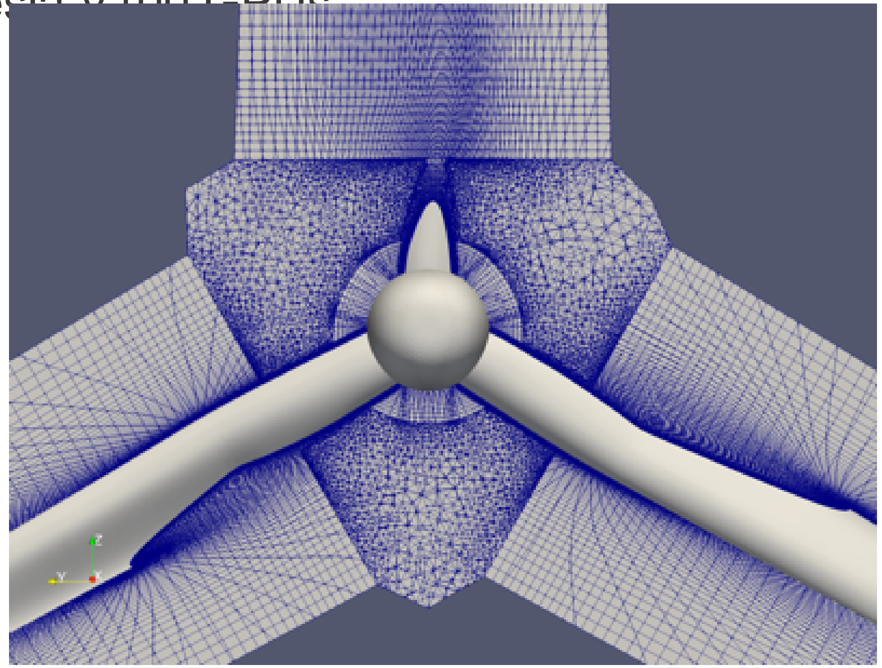
- Global size problem is fixed
- Decrease time-to-solution by adding compute resources

Two linear solves for initial wall distance

Simulation is run for 10 times steps.

- 4 Picard iterations per time step

40 linear solvers per physics phase



Numerical Experiments

Rotating wind turbine simulation using refined version of NREL5MW mesh

- 5MW reference wind turbine (Jonkman et al., NREL Tech Report #TP-500-38060, 2009)
- 634.5e6 nodes, 719.4e6 elements
- “hybrid” mesh: extruded structured around blade, unstructured around hub and hub/blade transition

Experiments run on ORNL Summit supercomputer

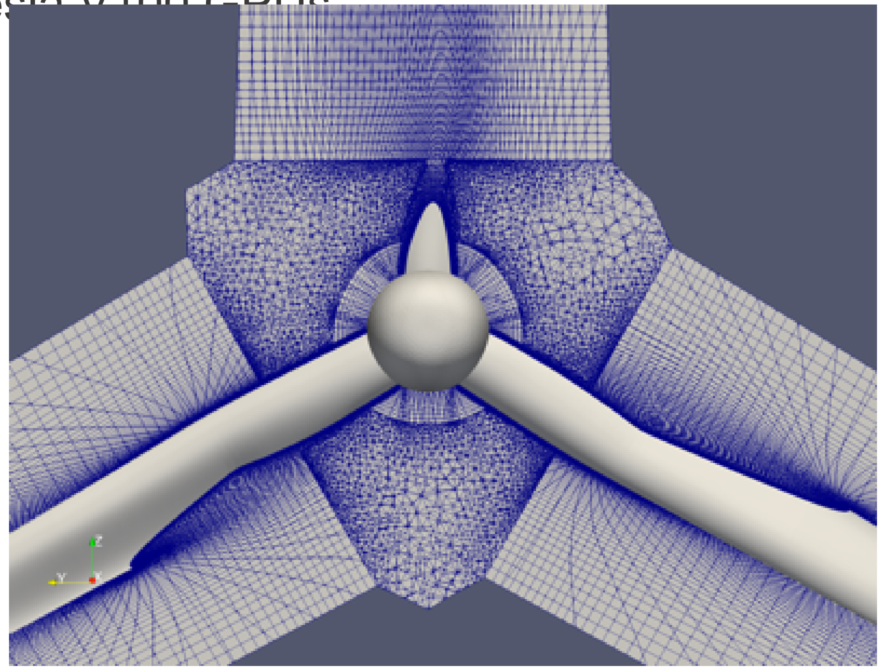
- 4600 compute nodes, each with two Power9 CPUs, six NVIDIA Tesla V100 GPUs

ExaWind is primarily interested in strong-scaling

- Global size problem is fixed
- Decrease time-to-solution by adding compute resources

Momentum linear solver: GMRES/SGS

Continuity linear solver: GMRES/AMG



Overall simulation wall clock times

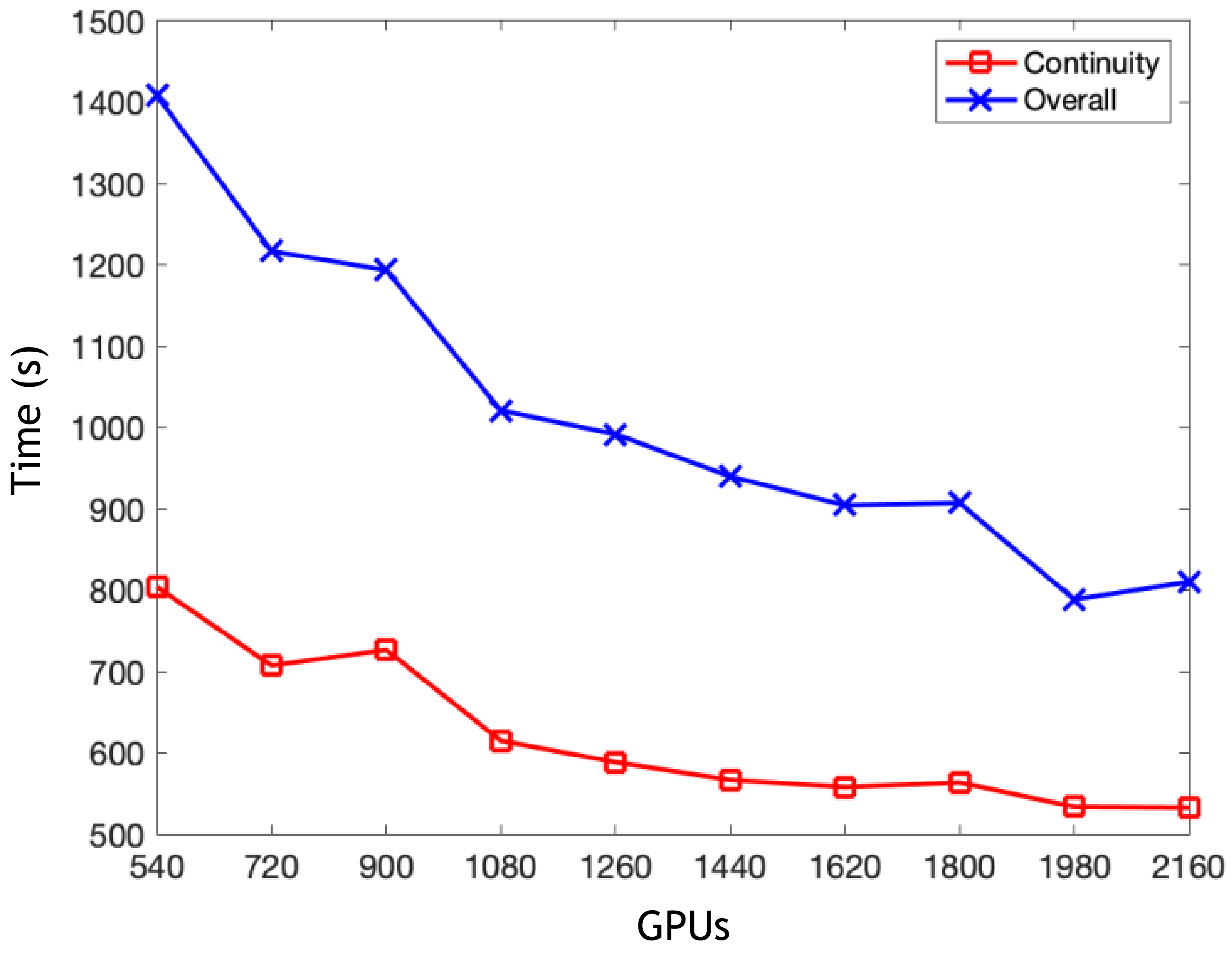
10 time steps

- 4 Picard iterations per time step

Momentum (not shown) is < 100s

Continuity phase accounts for > 50% of runtime

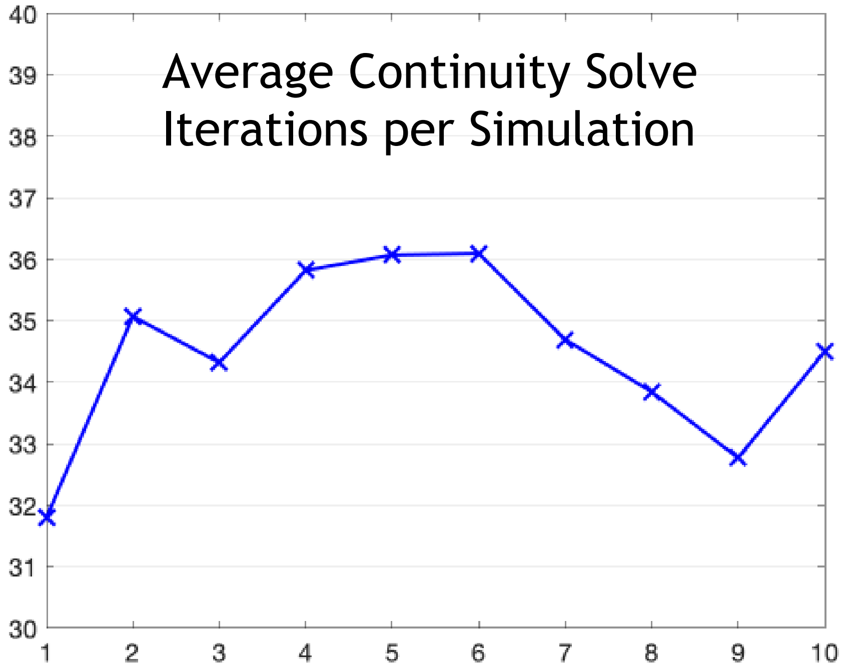
- Preconditioner setup and solve dominates

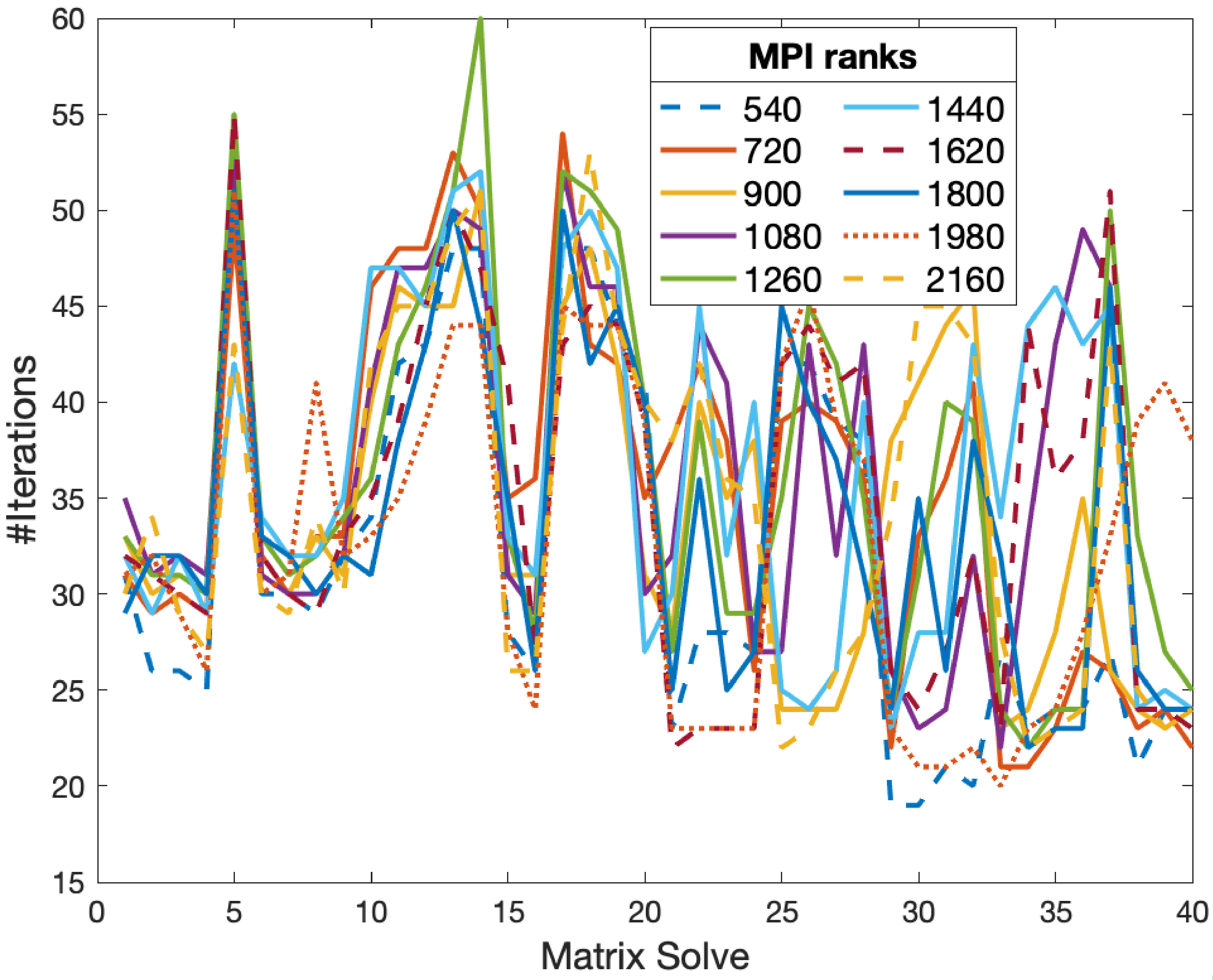


Continuity Solve: Algorithmic Scalability

Algorithmically scalable

- Some variability across individual solves





AMG solver details

Smoother: degree 2 Chebyshev polynomial

Coarse grid solve: degree 16 Chebyshev polynomial

Local ”greedy” aggregation, improve grid transfer via damped Jacobi iteration

Rebalancing of multigrid matrices to subset of GPUs

- Delayed until level 2 matrix or greater (level 0 = application matrix)
- Occurs if #rows per GPU falls below 10K

Multigrid Hierarchy @ 540 GPUs

level	rows	nnz	nnz/row	c ratio	procs
0	634469604	4652826078	7.33		540
1	73132340	2490326926	34.05	8.68	540
2	4687448	315661782	67.34	15.60	93
3	389076	37062352	95.26	12.05	7
4	29815	6095493	204.44	13.05	1

Operator complexity: 1.61

$$\text{Operator complexity} = \frac{\Sigma_{i=0}^{i=L} nnz(A_i)}{nnz(A_0)}$$

Multigrid Hierarchy @ 2160 GPUs

level	rows	nnz	nnz/row	c ratio	procs
0	634469604	4652826078	7.33		2160
1	73885977	2536703231	34.33	8.59	2160
2	4923757	346194059	70.31	15.01	98
3	404337	39939219	98.78	12.18	8
4	32311	7341495	227.21	12.51	1

Operator complexity: 1.63

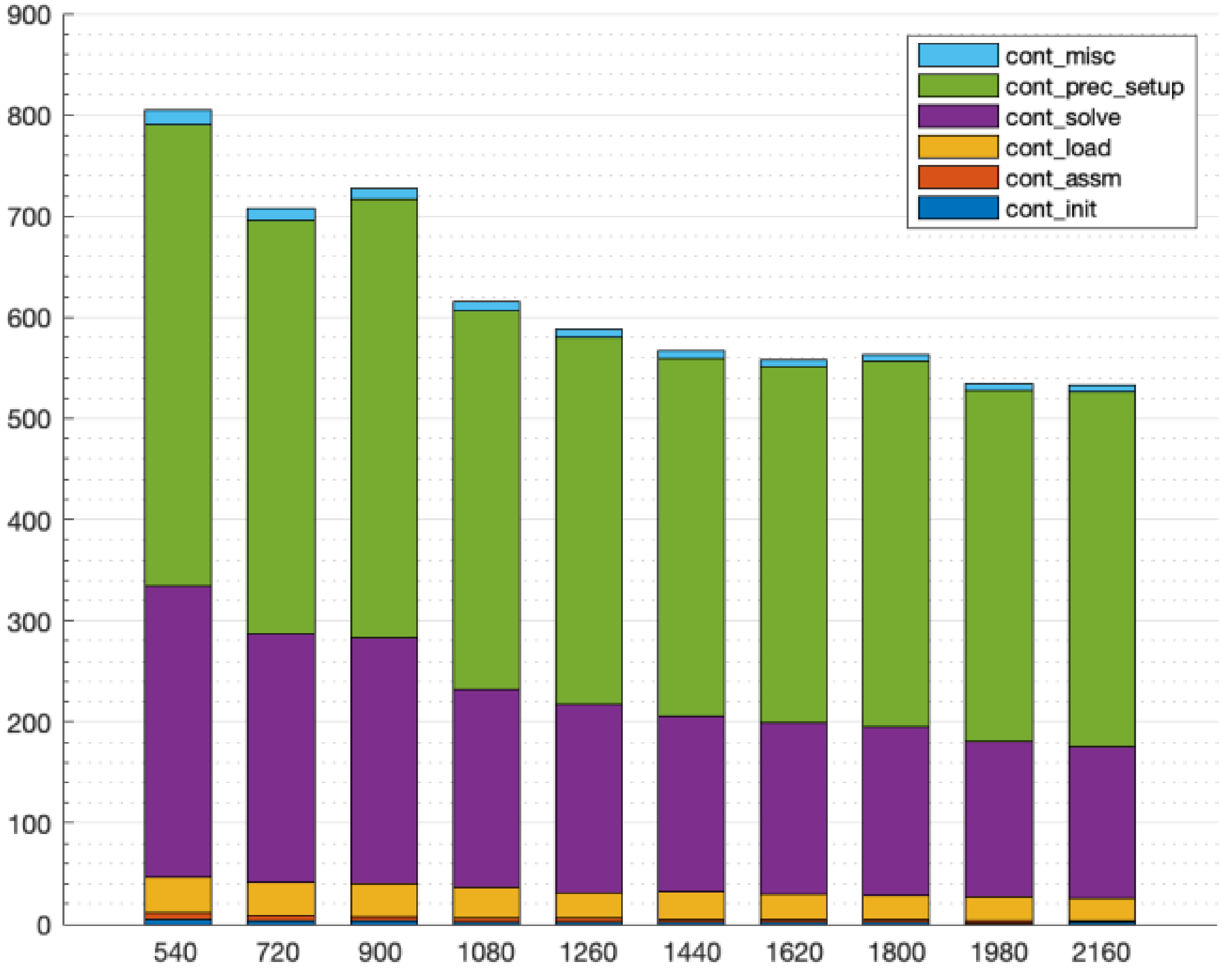
Continuity phase details

Matrix assembly

- “cont_load”, “cont_assembly”
- Negligible cost

Preconditioner setup and solve dominate

- Neither scales particularly well



Continuity: AMG Setup Time by Level

Level 0

- Application-supplied matrix
- smoother setup only

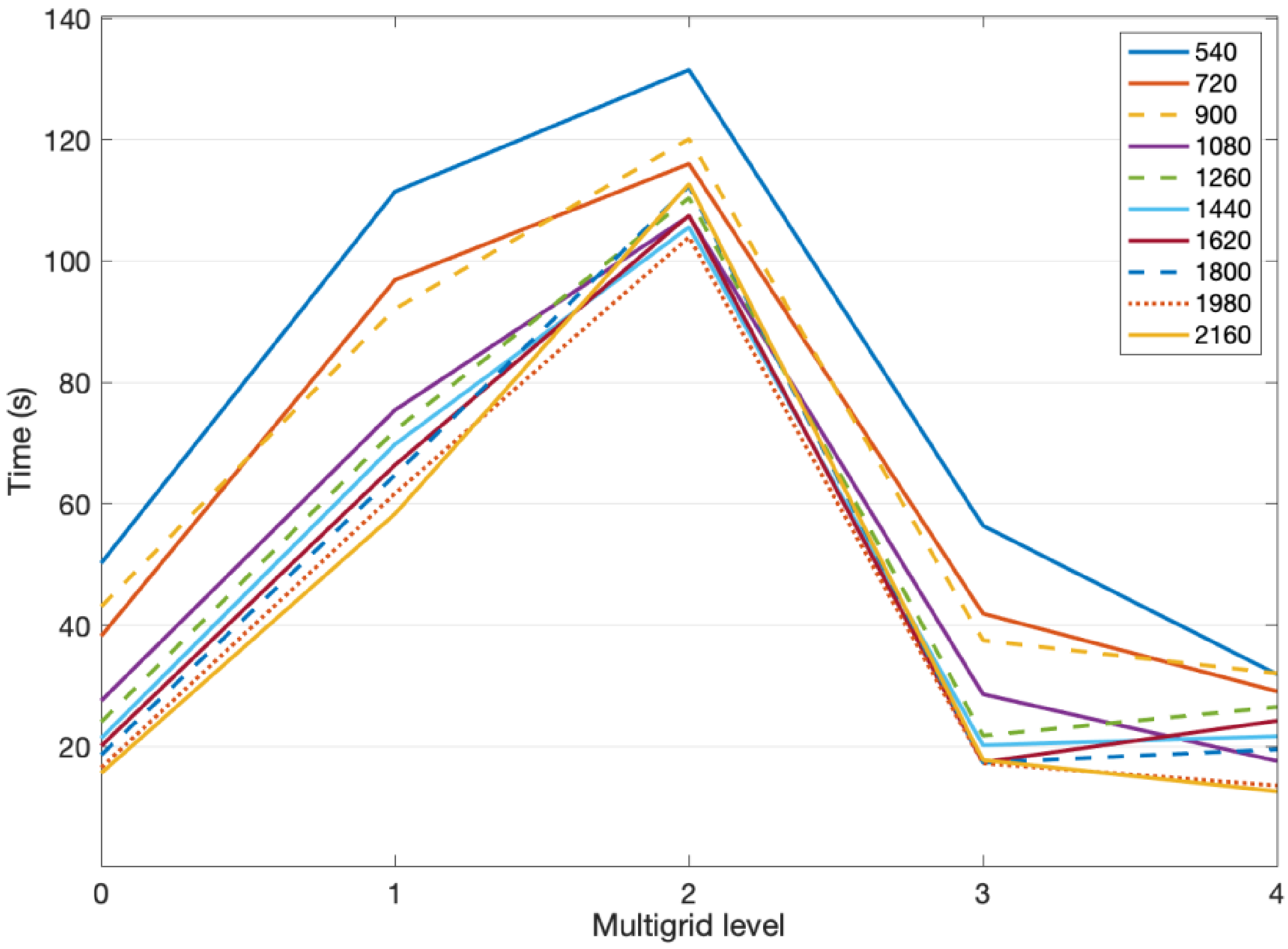
Level 1

- First coarse grid level

Level 2

- Rebalance matrix and move to subset of GPUs

Levels 3 and 4 are inconsequential



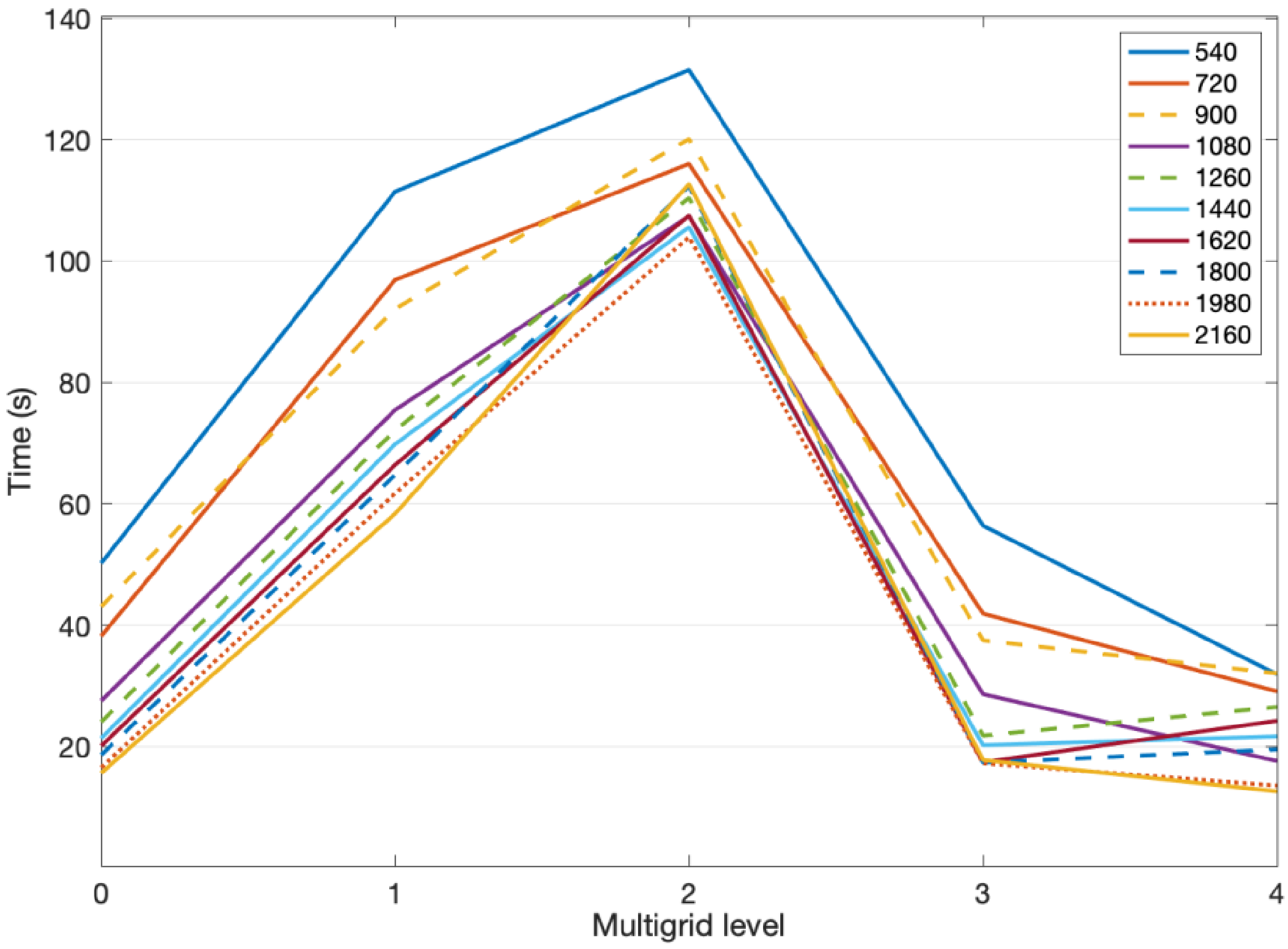
Continuity: AMG Setup Time by Level

Level 1

- @2160 GPUs
- Total: 58.3 s
  - Dropping weak connections: 20s
  - Triple-matrix product: 25s
  - Prolongator smoothing: 9s

Level 2

- @ 2160 GPUs
- Total: 112.7s
  - Dropping weak connections: 13s
  - Triple-matrix product: 32s
  - Transferring aux data: 24s
  - Rebalancing matrix: 25s



Some Observations

Chebyshev smoothing requires estimate of largest eigenvalue

- We've observed that 1.1 is a good estimate for all but coarsest system
- Avoids power iterations for eigen estimates

Unnecessary to coarsen to small system that can be solved directly

- Truncating hiearchy and applying iterative method to large coarse system is effective

An obvious next step is to optimize dropping of weak connections and auxiliary data transfers

Ongoing Work

Investigate balance between AMG setup and solve

- May be able to reduce AMG setup cost by creating lower complexity preconditioner
- Lowering complexity will likely hurt convergence

Refactor Nalu-Wind momentum linear system on GPU

- Currently (u,v,w) system
- Can be rewritten as scalar system with 3 right-hand sides
- Scalar matrix is 3x smaller than (u,v,w) matrix

Remove usage of uniform virtual memory (UVM) in Nalu-Wind itself

Assess performance on other ORNL platforms

Acknowledgments

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Funding was provided by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of two U.S. Department of Energy organizations (Office of Science and the National Nuclear Security Administration) responsible for the planning and preparation of a capable exascale ecosystem, including software, applications, hardware, advanced system engineering, and early testbed platforms, in support of the nation?s exascale computing imperative.