SAND2020-1256C

ExaWind: Exascale Predictive Wind Plant Flow Physics Modeling

NREL: M.A. Sprague (PI), S. Ananthan, M. Brazell, A. Glaws, M. Henry de Frahan, R. King, M. Natarajan, J. Rood, A. Sharma, K. Swirydowicz, S. Thomas, G. Vijayakumar, S. Yellapantula

SNL: P. Crozier, L. Berger-Vergiat, L. Cheung, D. Glaze, J. Hu, R. Knaus, D. Lee, T. Okusanya, J. Overfelt, S. Rajamanickam, P. Sakievich, T. Smith, J. Vo, A. Williams, I. Yamazaki

ORNL: J. Turner, A. Prokopenko, R. Wilson

UTA: R. Moser, J. Melvin

Parallel Geometric Algorithms: Jay Sitaraman

EXASCALE COMPUTING PROJECT

Project Overview

Objective: Create a predictive physics-based simulation capability that will provide a validated "ground truth" foundation for wind plant siting, operational controls, and reliably integrating wind energy into the grid

Motivation: Validated, predictive wind plant simulations will reduce the cost of energy by providing

- a path to better understanding of wind plant flow physics, which will lead to
  - new plant layout design in complex terrain
  - new turbine technologies to optimize plant performance
- a foundation for improved computer-aided engineering models, which will enable better design optimization

Primary Application Codes:

- **Nalu-Wind**
  - https://github.com/exawind/nalu-wind
  - Unstructured-grid computational fluid dynamics (CFD) code
  - Based on the SNL-supported Nalu code
  - C/C++
  - Built on Trilinos/STK/hypre/TIOGA
- **OpenFAST**
  - https://github.com/openfast/openfast
  - Whole-turbine simulation code; blades, control system, tower, etc.
  - Fortran 90; dedicated Intel Parallel Computing Center (IPCC) for parallelization

Software/Library Partnerships in Nalu-Wind

- ***Trilinos***, https://trilinos.org/
  - ***MueLu:*** provides aggregation-based multigrid preconditioners
  - ***Ifpack2:*** provides SOR-based, polynomial and incomplete factorization preconditioners
  - ***Kokkos-Kernels:*** provides shared memory algorithms: graph-coloring, SpMV, SPMM, iterative and incomplete factorization preconditioners
  - ***Tpetra:*** provides distributed memory, sparse linear algebra objects
  - ***Belos:*** provides templated Krylov and recycling solvers
  - ***Amesos2:*** provides sparse direct solvers
  - ***Sierra Toolkit (STK):*** provides an unstructured-mesh in-memory, parallel-distributed database
- ***hypre*,** https://github.com/LLNL/hypre
  - Multigrid solvers and preconditioners based on classic Ruge-Stüben AMG algorithm
- ***Kokkos***, https://github.com/kokkos
  - Programming model in C++ for writing performance portable applications targeting all major HPC platforms
- ***TIOGA*,** https://github.com/jsitaraman/tioga
  - Library for overset-grid assembly on parallel distributed systems
- ***VTK-m*,** https://gitlab.kitware.com/vtk/vtk-m
  - New *in situ* visualization and analysis capabilities
- **Spack,** https://github.com/spack/spack
  - Package manager for exascale software
- **AMReX,** https://github.com/AMReX-Codes/amrex
  - Software Framework for Block Structured AMR

ECP Key Performance Parameter (KPP-2)

Challenge Problem:

Predictive simulation of a wind farm with tens of megawatt-scale wind turbines dispersed over an area of 50 square kilometers

Minimum Requirements:

- 3x3 array of megawatt-scale turbines operating at rated speed
- 4 km x 4 km domain with height of 1 km
- Hybrid-RANS/LES model
- At least 30-billion gridpoints (and 150 billion degrees of freedom); near-blade grid spacing will be such that the viscous sub-layer is resolved
- Demonstrate that we can simulate at least one domain transit time (500 s) with four weeks of system time

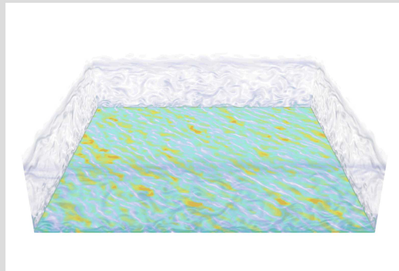
Keys to Success

- Enable large time steps (restricted by accuracy rather than stability)
- Minimize time per timestep:
  - Optimize strong scaling to utilize as much of the system as possible
  - Optimize linear-system solver algorithms

2019 Highlights

Established structured-grid background solver amr-wind

- amr-wind: Forked from incompressible-flow Navier-Stokes solver incflo
- Utilizes LBNL AMReX framework
- Includes newly implemented neutral ABL physics
- Performance portable and GPU ready
- Successful collaboration with the LBNL AMReX team

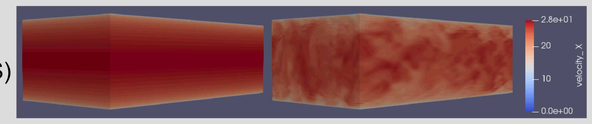


3x3x1 km neutral atmospheric boundary layer (ABL) simulation, contours of vorticity.

Team: Almgren (LBNL), *Ananthan, Brazell, Cheung, Natarajan, Rood, Sakievich, Yellapantula,* Zhang (LBNL)

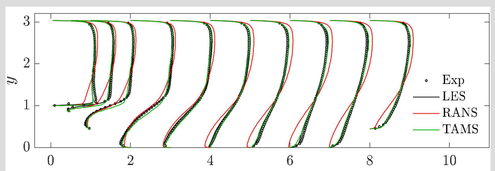
Validated a new hybrid RANS/LES turbulence model for wind turbine simulations

- Deployed Time-Averaged Model Split (TAMS) Hybrid RANS/LES framework and implemented it in the Nalu-Wind solver stack



$Re_\tau = 5200$ *Stationary Channel Flow model verification test in Nalu-Wind. Left: RANS solution, Right: TAMS solution. The TAMS solution is able to generate turbulent fluctuations without the typical log-layer mismatch seen in other hybrid RANS/LES models.*

- Implemented TAMS model framework in Nalu-Wind and performed model verification tests; initial tests of TAMS conducted on NACA-0015 fixed wing and NREL 5MW turbine in atmospheric flow
- Next steps will be focused on resolving numerical challenges related to TAMS implementation in Nalu-Wind, such as relaxation of restrictive timesteps in turbine simulations due to explicit treatment of one of the TAMS model terms

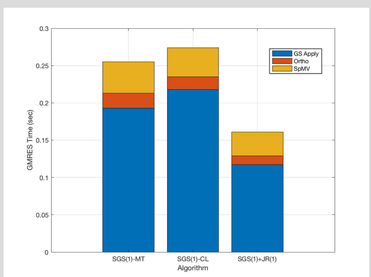


Validation of the TAMS Hybrid model on a canonical periodic hill. This test requires the capture of smooth wall separation and reattachment. Comparison to LES and experimental results shows excellent agreement, while utilizing 1/40th of the needed grid points relative to wall-resolved LES.

Team: *Melvin, Henry de Frahan, Vijayakumar, Ananthan, Moser*

SpMV outperforms tri-solve on GPUs: Trilinos & *hypre*

- Traditional symmetric Gauss-Seidel (SGS) with multi-threading (MT), and clustering (CL), versus two-stage SGS with Jacobi-Richardson (JR) inner iteration
  - *SGS-MT from Rajamanickam and Deveci*
  - *SGS-CL from Kelly and Rajamanickam*
- Coloring triangular-solve ordering can increase the number of GMRES iterations, increasing time to solution.
- SGS(1) – JR(1) *reduces* iteration count
- Kokkos kernels and CUDA
- Sparse matrix-vector multiply SpMV replaces triangular solver recurrences



Nalu-Wind momentum GMRES Trilinos solver times on one GPU for the McAlister blade problem, with 3.2M DOF; compares SGS-MT, SGS-CL and SGS Jacobi-Richardson nested SGS preconditioners; SGS(1)-JR(1) leads to 50% improvement

Team: Thomas, Yamazaki , Swirydowicz, Berger-Vergiat, Hu

Nalu-Wind simulations on Summit using Trilinos

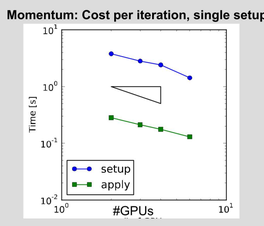
Strong-scaling: single-node GPU
Neutrally stable atmospheric boundary layer (ABL) simulation; rectangular domain with structured mesh and 3.1M elements

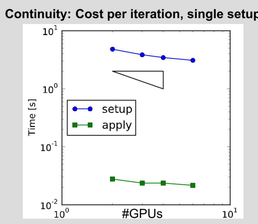
Momentum solver

- GMRES+Gauss-Seidel
- Coupled/monolithic (u,v,w) solve
- `Setup` & `Apply` scale almost perfectly

Continuity solver

- GMRES+SA-AMG
- `Apply` is fast, but needs improvement in order to strong scale
- `Setup` is work-in-progress





Weak-scaling: multi-node GPUs

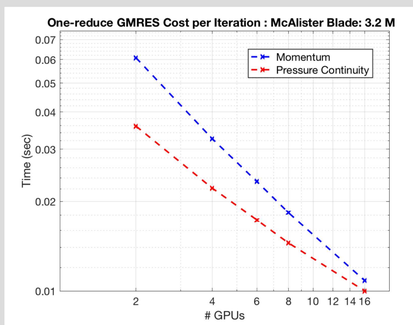
- Nalu-Wind ABL simulation with Trilinos has been run up to 1000 GPUs for mesh containing 1.25B elements
- Initial results show linear systems account for ~75% of simulation time
  - Reducing total time spent on linear systems is critical to allow effective utilization of Summit and future exascale supercomputers

ABL simulation runs end-to-end on GPUs with Trilinos

- New computational capability
- First-ever weak-scaling results up to 1K GPUs
- Assembly using STK/Tpetra/Kokkos
- Linear solvers using Belos/Ifpack2/MueLu/Kokkos-Kernels/Zoltan2
- Kokkos ensures portability to new machines like Frontier and Aurora

Team: Berger-Vergiat, Ananthan, Williams, Hu, Rajamanickam, Yamazaki

GMRES one-reduce scalable solver: Nalu-Wind on GPU



Solver time per iteration for the McAlister blade problem with 3.2M DOF; segregated (u,v,w) system solver; V100 GPUs on NREL Eagle HPC; two-stage smoother is at least 2x faster than tri-solve GS smoother

- *hypre* GMRES one-reduce solvers achieve O(0.01) sec per iteration
- 16 Intel Skylake CPU cores (ranks) give similar iteration time as 2 GPUs
- Solver described in *Swirydowicz et al. (2019) Low-synch Gram-Schmidt and GMRES Algorithms*

Team: Swirydowicz, Thomas, Ananthan, Sprague, Langou (CU), Bielich (CU)

Solvers team reduces solve time on GPU for Nalu-Wind pressure and momentum systems with overset meshes

- Pressure and momentum system setup & solve times reduced to less than 30% from 75% of total time (for McAlister-blade problem)
- Segregated momentum solver accelerated 2x with two-stage smoother
- Improvements were demonstrated in both the *hypre* and Trilinos sparse linear-system solver stacks
- Details in
  - Thomas et al., Incompressible Navier-Stokes with Two-Stage Gauss-Seidel Smoothers. *Copper Mountain Conference on Iterative Methods 2020, invited talk* (to be presented)
  - Thomas et al (2020), Two-Stage Gauss-Seidel Smoothers, *SIAM Conference on Parallel Processing for Scientific Computing 2020,*
  - Yamazaki et al (2020), s-step and pipelined Krylov solvers on GPUs with Trilinos. *SIAM Conference on Parallel Processing for Scientific Computing 2020*

Team: Thomas, Swirydowicz, Yamazaki, Ananthan, Hu, Sprague

This research was supported 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.

Exascale Computing Project Annual Meeting
Houston, TX
February 4-7, 2020