SAND2018-1807R

ECP Milestone Report

Deploy production sliding mesh capability with linear solver benchmarking

WBS 2.2.2.01, Exawind Milestone ECP FY18 Q1

Stefan Domino, Stephen Thomas, Matt Barone, Alan Williams, Shreyas Ananthan, Robert Knaus, James Overfelt, Mike Sprague, and Jon Rood

20 December 2017

DOCUMENT AVAILABILITY

Reports produced after January 1, 1996, are generally available free via US Department of Energy (DOE) SciTech Connect.

Website http://www.osti.gov/scitech/

Reports produced before January 1, 1996, may be purchased by members of the public from the following source:

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
Telephone 703-605-6000 (1-800-553-6847)
TDD 703-487-4639
Fax 703-605-6900
E-mail info@ntis.gov
Website http://www.ntis.gov/help/ordermethods.aspx

Reports are available to DOE employees, DOE contractors, Energy Technology Data Exchange representatives, and International Nuclear Information System representatives from the following source:

Office of Scientific and Technical Information
PO Box 62
Oak Ridge, TN 37831
Telephone 865-576-8401
Fax 865-576-5728
E-mail reports@osti.gov
Website http://www.osti.gov/contact.html

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ECP Milestone Report
Deploy production sliding mesh capability with linear solver benchmarking
WBS 2.2.2.01, Milestone ECP FY18 Q1

Office of Advanced Scientific Computing Research
Office of Science
US Department of Energy

Office of Advanced Simulation and Computing
National Nuclear Security Administration
US Department of Energy

20 December 2017

ECP Milestone Report
Deploy production sliding mesh capability with linear solver benchmarking
WBS 2.2.2.01, Milestone ECP FY18 Q1

Approvals

Submitted by:

Stefan P. Domino

20 Dec 2017

Stefan Domino (SNL)
ECP FY18 Q1

Date

Approval:

Andrew Siegel, Argonne National Laboratory
Director, Applications Development
Exascale Computing Project

Date

Revision Log

Version	Creation Date	Description	Approval Date
1.0	20 December 2017	Original	
1.1	22 February 2018	Revision based on comments from Tom Evans	

EXECUTIVE SUMMARY

Wind applications require the ability to simulate rotating blades. To support this use-case, a novel design-order sliding mesh algorithm has been developed and deployed. The hybrid method combines the control volume finite element methodology (CVFEM) with concepts found within a discontinuous Galerkin (DG) finite element method (FEM) to manage a sliding mesh. The method has been demonstrated to be design-order for the tested polynomial basis (P=1 and P=2) and has been deployed to provide production simulation capability for a Vestas V27 (225 kW) wind turbine. Other stationary and canonical rotating flow simulations are also presented. As the majority of wind-energy applications are driving extensive usage of hybrid meshes, a foundational study that outlines near-wall numerical behavior for a variety of element topologies is presented. Results indicate that the proposed nonlinear stabilization operator (NSO) is an effective stabilization methodology to control Gibbs phenomena at large cell Peclet numbers. The study also provides practical mesh resolution guidelines for future analysis efforts. Application-driven performance and algorithmic improvements have been carried out to increase robustness of the scheme on hybrid production wind energy meshes. Specifically, the Kokkos-based Nalu `Kernel` construct outlined in the FY17/Q4 ExaWind milestone has been transitioned to the hybrid mesh regime. This code base is exercised within a full V27 production run. Simulation timings for parallel search and custom ghosting are presented. As the low-Mach application space requires implicit matrix solves, the cost of matrix reinitialization has been evaluated on a variety of production meshes. Results indicate that at low element counts, i.e., fewer than 100 million elements, matrix graph initialization and preconditioner setup times are small. However, as mesh sizes increase, e.g., 500 million elements, simulation time associated with “setup-up” costs can increase to nearly 50% of overall simulation time when using the full Tpetra solver stack and nearly 35% when using a mixed Tpetra-Hypre-based solver stack. The report also highlights the project achievement of surpassing the 1 billion element mesh scale for a production V27 hybrid mesh. A detailed timing breakdown is presented that again suggests work to be done in the setup events associated with the linear system. In order to mitigate these initialization costs, several application paths have been explored, all of which are designed to reduce the frequency of matrix reinitialization. Methods such as removing Jacobian entries on the dynamic matrix columns (in concert with increased inner equation iterations), and lagging of Jacobian entries have reduced setup times at the cost of numerical stability. Artificially increasing, or bloating, the matrix stencil to ensure that full Jacobians are included is developed with results suggesting that this methodology is useful in decreasing reinitialization events without loss of matrix contributions. With the above foundational advances in computational capability, the project is well positioned to begin scientific inquiry on a variety of wind-farm physics such as turbine/turbine wake interactions.[1]

[1] Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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. This report followed the Sandia National Laboratories formal review and approval process (SAND2017-13716 R). As such, the technical report is suitable for unlimited release.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Many applications in computational fluid dynamics require predicting a complex flow field in the presence of moving boundaries. For example, in turbomachinery, rotorcraft, and wind energy applications, a solid boundary must be rotated within a turbulent flow. The goal of this milestone is to develop, verify, deploy, and benchmark a novel hybrid CVFEM/DG discretization approach for wind energy applications that involve rotating blades.

The managing of moving boundaries is not without complexity as this use case requires geometric mesh connectivity changes. This dynamic mesh connectivity drives algorithmic requirements including parallel searches, matrix graph re-initialization, and frequent preconditioner setup events. In most cases previously run, the cost of such re-initialization events could be amortized over the full simulation time as the meshes were static in nature. However, for the end ExaWind use case, which requires the simulation of O(100) multi-MW wind turbines sited within a 10 km x 10 km domain, mesh movement will be routine. Moreover, billions of computational elements will be required. In support of exascale computing, scaling of these core interfaces will be critical to the success of the project.

In section 2, the milestone description is provided. Section 3 provides a high-level description of the hybrid numerical methodology to support the sliding mesh use case. A small sampling of code verification that has been conducted is also provided. A set of production runs is provided in section 4. Solver and application performance benchmarking/scaling is provided in section 5. Application improvements including methods to reduce the frequency or effectiveness of matrix re-initialization is presented in section 6. Finally, in section 7, notable achievements, conclusions and an identified path forward is described.

2. MILESTONE DESCRIPTION & REQUIREMENTS

2.1 Milestone Description:

Although a preliminary sliding mesh/overset method has been developed in the Nalu application code base, improvements in algorithmic robustness in the presence of surface faceting must be accomplished. Moreover, efficient parallel searches (leveraged from the SNL-ATDM effort) must be deployed. Evaluation and possible incorporation of load balancing techniques, either in situ or as a pre-processing Seacas::decomp() step, should be explored should the search/assemble procedure warrant improvements. Finally, matrix initialization performance at the desired early project scale, i.e., O(1) billion elements, should be established to guide future code infrastructure improvements. Where possible, application modifications to reduce the frequency of setup costs will be evaluated and deployed.

2.2 Milestone Execution Plan:

1. Improve baseline sliding mesh capability at curved surfaces.
2. Evaluate ATDM-based parallel search methods.
3. Establish matrix set-up cost timings.
4. Evaluate possible lagging of matrix update.
5. Evaluate reduction of matrix system by omitting moving block column entries in favor of multiple matrix assembly/solve iterations.

2.3 Milestone Completion Criteria:

A report will be created documenting evaluated and executed algorithm design, and simulation timings along with any identified bottlenecks. Code created will be posted on the public repository along with relevant documentation.

3. NUMERICAL OVERVIEW

The core ExaWind numerical methodology falls within the class of vertex-centered finite volume schemes with specific emphasis on the control volume finite element method (CVFEM) [11]. For more information on the core algorithm, the reader is referred to the Nalu theory document [5], or recent low- and higher-order descriptions [4]. The suite of numerical methods under consideration can be found in [3].

3.1 Dual Volume Definition

The control volume finite element method simply defines a dual mesh constructed within each element. A weak variational statement is written and a piece-wise constant test function is applied. Figure 1 provides an overview of both low-order and higher-order node and dual volume rules. Integration points are defined at two locations: for flux calculations, the method uses subcontrol surfaces; for source and time contributions, the subcontrol volume center of each dual volume is used. To recover a low-order edge-based vertex-centered (EBVC) method, dual volume subcontrol area vectors are assembled to the edges of the low-order elements while dual nodal volumes are assembled to the nodes of the elements. Both the edge- and element-based schemes can be considered a Petrov-Galerkin method.

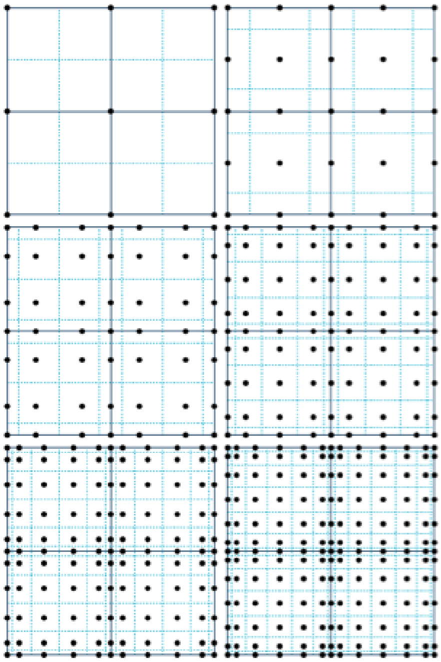


Figure 1: Polynomial promotion for a canonical CVFEM quad element patch from P=1 (upper left) to P=6 (lower right).

3.2 Hyrbid CVFEM/DG Sliding Mesh Algorithm

In wind energy applications, both sliding mesh and overset methods are viable approaches to manage the moving body fluid interaction. Although both are actively developed in ExaWind, the sliding mesh capability is highlighted in this milestone report. In sliding mesh applications, a prescribed mesh interface is defined to manage the solid body movement/rotation. Recently, a design-order hybrid CVFEM/DG scheme has been developed and deployed to the wind energy application space [6]. In this section, only a high-level description of the scheme will be provided. More details can also be found in the Nalu online theory manual [2].

The description of the methodology begins as follows: consider two domains, Ω^A and Ω^B, which have a common interface, Γ^{AB}, and a set of interfaces not in common, $\Gamma \backslash \Gamma^{AB}$, see Figure 2. Each domain has a set

[2] http://nalu.readthedocs.io

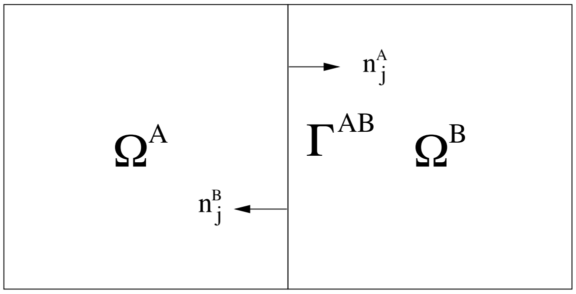


Figure 2: Two-block example with two blocks, Ω^A and Ω^B, and one common surface definition, Γ^{AB}.

of outwardly pointing normals n_j^A and n_j^B. In the idealized case, these normals are perfectly opposite from each other. In practice, due to surface faceting, such an idealized interface will not be found at low-order curved surfaces.

The objective of the sliding mesh algorithm is to provide a design-order reconstruction technique to manage the non-conformal configuration. Principles of DG are used to define an interior penalty approach [1]. As a high-level example, consider the time varying heat conduction equation for temperature, T, on each block Ω^A and Ω^B with a nonconformal interface as defined in Figure 2:

$$\begin{aligned}\int_{\Omega^A} w\left(\rho C_p \frac{\partial T}{\partial t} - S\right) d\Omega \quad &- \int_{\Omega^A} \frac{\partial w}{\partial x_j} q_j d\Omega \\ &+ \int_{\Gamma \backslash \Gamma^{AB}} w q_j n_j d\Gamma \\ &+ \int_{\Gamma^{AB}} w\hat{T}(A,B) d\Gamma \\ &+ \int_{\Gamma^{AB}} \frac{\partial w}{\partial x_j} n_j^A \lambda_{IP}^{AB} (T^A - T^B) d\Gamma, \end{aligned} \tag{1}$$

$$\begin{aligned}\int_{\Omega_B} w\left(\rho C_p \frac{\partial T}{\partial t} - S\right) d\Omega \quad &- \int_{\Omega_B} \frac{\partial w}{\partial x_j} q_j d\Omega \\ &+ \int_{\Gamma \backslash \Gamma_{BA}} w q_j n_j d\Gamma \\ &+ \int_{\Gamma_{BA}} w\hat{T}(B,A) d\Gamma \\ &+ \int_{\Gamma_{BA}} \frac{\partial w}{\partial x_j} n_j^B \lambda_{IP}^{BA} \left(T^B - T^A)\right) d\Gamma. \end{aligned} \tag{2}$$

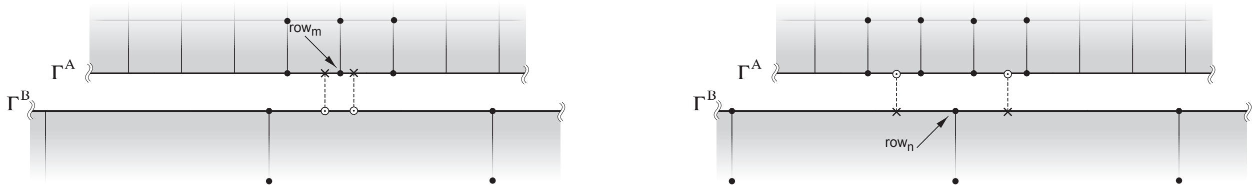
Above, ρ and C_p represent density and specific heat, respectively. The heat flux diffusional vector, q_j, is given by,

$$q_j = -k \frac{\partial T}{\partial x_j}. \tag{3}$$

The penalty parameter, λ and numerical flux, $\hat{T}(\alpha, \beta)$ define the choice of the DG scheme. For piecewise constant CVFEM test functions, the final form of the above equation set is as follows:

$$\int_{\Omega^A} \left(\rho C_p \frac{\partial T}{\partial t} - S\right) d\Omega + \int_{\Gamma \backslash \Gamma^{AB}} q_j n_j d\Gamma + \int_{\Gamma^{AB}} \hat{T}(A,B) d\Gamma = 0, \tag{4}$$

$$\int_{\Omega^B} \left(\rho C_p \frac{\partial T}{\partial t} - S\right) d\Omega + \int_{\Gamma \backslash \Gamma^{BA}} q_j n_j d\Gamma + \int_{\Gamma^{BA}} \hat{T}(B,A) d\Gamma = 0. \tag{5}$$



(a) Current and opposite sides are A and B respectively.

(b) Current and opposite sides are B and A respectively.

Figure 3: Description of the numerical flux calculation for the set of current and opposite interfaces.

The numerical flux is defined as,

$$\hat{T}(\alpha, \beta) = \frac{1}{2}(q_j^\alpha n_j^\alpha - q_j^\beta n_j^\beta) + \lambda_T^{\alpha\beta}(T^\alpha - T^\beta). \tag{6}$$

Note that the interior penalty term, i.e., the gradient of the test function vanishes in the CVFEM scheme.

Figure 3 graphically demonstrates the procedure in which current integration point locations for a set of non-conformal surfaces are projected to the opposing exposed non-conformal face. The methodology follows a dual-pass approach. First, the current face exposed integration points are projected to the opposing surface. An integration point opposing face/element data structure is defined to capture the required mesh connectivity. In the second pass, the former opposing exposed faces, which are now the current exposed integration point set, are projected to the opposing face.

Matrix contributions are desired to be provided for both the current and opposing element connectivity. The given mesh decomposition will likely require parallel searches to obtain the opposing integration point elemental contributions. Moreover, the connectivity between the exposed face and attached element, which is obtained by the Sierra Toolkit, is required to adequately specify the column entries for each row on the exposed surface.

The required extensions to a low-Mach formulations are described in the theory manual with details appearing in the recently submitted JCP article [6].

3.3 Hybrid CVFEM/DG Verification

In order to provide a high-level overview of code verification, two cases will be provided. The first is a simple steady Laplace operator while the second is a viscous vortex temporal case. Both have analytical solutions that are provided elsewhere [6]. Each study provides the algorithmic order-of-accuracy for P=1 and P=2 CVFEM approaches. Although the code has been transitioned to use arbitrary polynomial promotion, this study is limited to quadratic hexahedral-based elements.

Figure 4 provides the functional form for the manufactured scalar temperature solution while Figure 5 provides order-of-accuracy for a second- and third-order scheme. As with previous verification efforts, even with modest polynomial promotion, significant error reductions are noted.

In the second verification case highlighted in this report, a viscous vortex, which originates either in a P=1 or P=2 mesh block, travels past a non-conformal interface to either a P=1 or P=2 mesh block. This use case is germane to the wind energy application space as it is anticipated that a hybrid polynomial order will be in use, e.g., low-order near the blade and high-order within the wake. Figure 6 provides shadings of the velocity magnitude at the initial and final time plane while Figure 7 outlines the velocity error norms at 0.1 seconds. As can be seen from the norm calculations, the homogeneous higher-order scheme provides the best accuracy. As shown, the limiting numerical accuracy (P=1) prevails over each hybrid mesh block. Effectively, error in the first P=1 block is carried through to the P=2 block just as the high-accuracy P=2 solution is corrupted by the lower-order P=1 block. Studies in mixed-order methods within the wind energy application space are in order (see the FY17/Q3 ExaWind validation of a tip vortex [10] as a good candidate).

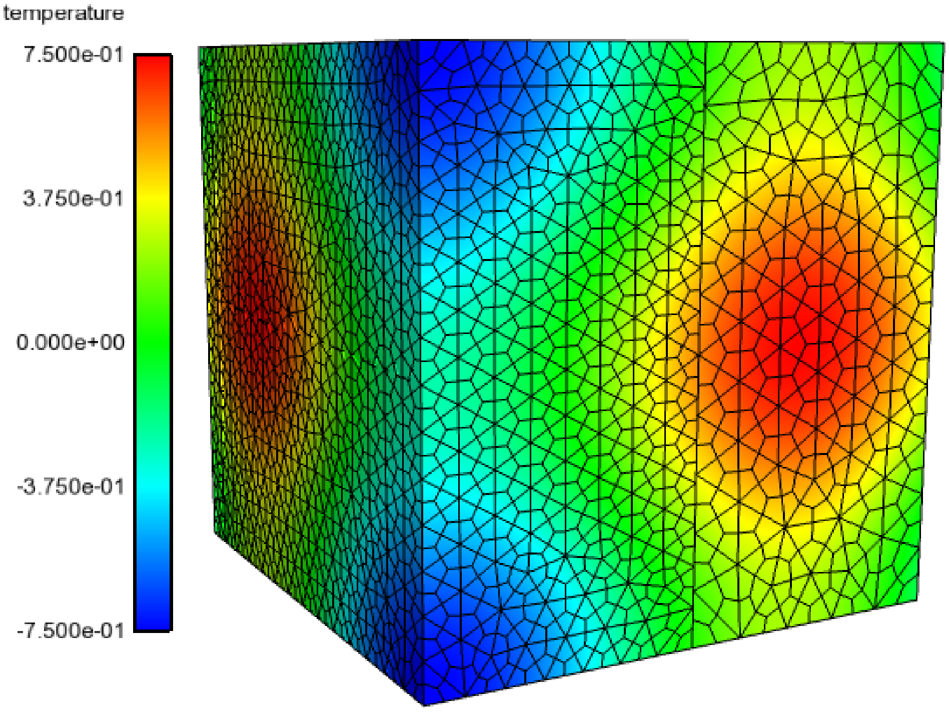


Figure 4: Steady, thermal diffusion manufactured solution; shown are shadings of the temperature field.

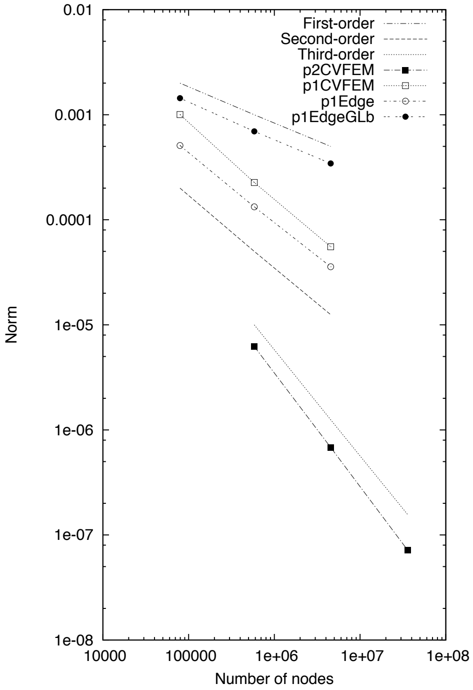
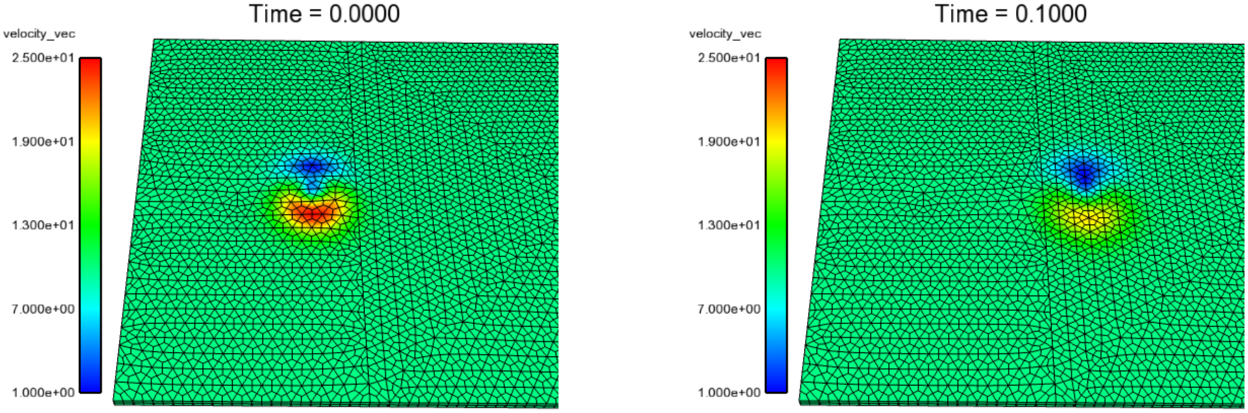


Figure 5: Steady, Laplace order of accuracy.



(a) Time = 0.0 seconds

(b) Time = 0.10 seconds.

Figure 6: Viscous vortex velocity magnitude shadings at initial and final time planes.

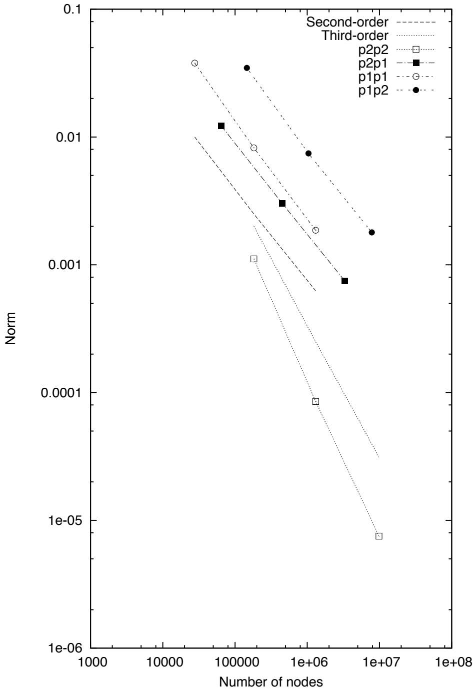


Figure 7: Viscous vortex norms for varying polynomial order combinations at the non-conformal interface.

3.4 Nonlinear Stabilization Operator

Although the deployed numerical scheme in Nalu can be classified as a low-dissipation scheme, which generally relies on central or Galerkin-based advection operators, at extreme cell Peclet numbers, a stabilization approach is required to adequately control the Gibbs phenomenon. This numerical behavior is simply defined as numerical ringing due to the non-monotonicity of the combined advection/diffusion operator. Although the code supports flux reconstruction methods (with classic limiters such as the Van Leer approach), such methods, although effective, are numerically diffuse. Moreover, they are (in most cases) limited to second-order spatial accuracy on unstructured meshes. Upwinded residual-based approaches [12] are also coded in Nalu.

In this project, extensive research concentrated on stable and robust stabilization operators is anticipated. Towards this end, the current numerical stabilization approach has been to combine elements of the entropy-viscosity method [8] with a discontinuity capturing operator (DCO) [12] found in the high-speed compressible FEM literature. In this method, a non-isotropic diffusional flux coefficient is proportional to a fine-scale residual. The fine-scale residual can be based on the degree-of-freedom itself or a fine-scale turbulent kinetic energy residual. This stabilization kernel is design-order and, as will be shown, has proven to be an effective stabilization approach when used in concert with the standard low-dissipation operators.

Although there are out-year project deliverables to drive the continued development and improvement of the NSO, the stabilization kernel currently in use for any variable ϕ is simply defined as,

$$-\sum_e \int_\Gamma \nu(\mathbf{R}) g^{ij} \frac{\partial \phi}{\partial x_j} n_i dS. \tag{7}$$

A fourth-order form, which writes the stabilization as the difference between the Gauss-point gradient and the projected nodal gradient interpolated to the Gauss-point, is also supported,

$$-\sum_e \int_\Gamma \nu(\mathbf{R}) g^{ij} (\frac{\partial \phi}{\partial x_j} - G_j \phi) n_i dS. \tag{8}$$

For completeness, the covariant and contravarient metric tensor are given by,

$$g^{ij} = \frac{\partial x_i}{\partial \xi_k} \frac{\partial x_j}{\partial \xi_k}, \tag{9}$$

and

$$g_{ij} = \frac{\partial \xi_k}{\partial x_i} \frac{\partial \xi_k}{\partial x_j}, \tag{10}$$

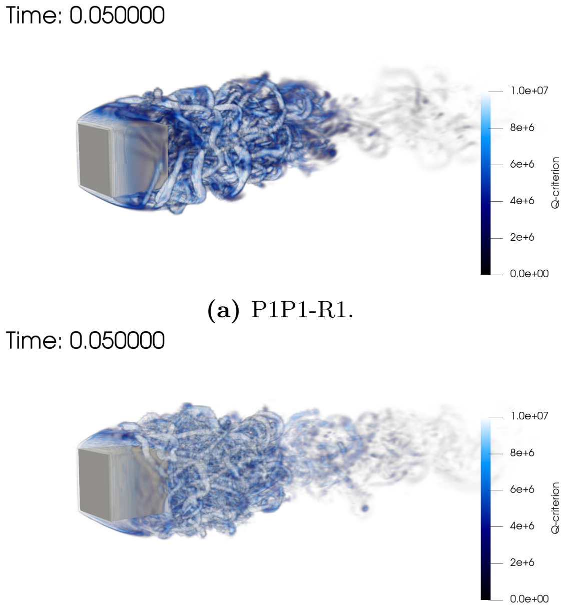
where $\xi = (\xi_1, \xi_2, \xi_3)^T$. The current form of $\nu(\mathbf{R})$ currently used is as follows,

$$\nu = \sqrt{\frac{\mathbf{R_k R_k}}{\frac{\partial \phi}{\partial x_i} g^{ij} \frac{\partial \phi}{\partial x_j}}}. \tag{11}$$

In the above expression for the non-isotropic viscosity ν, the residual, $\mathbf{R_k}$ can be approximated in many ways. The residual ($\mathbf{R_k}$) can be based on the fine-scale partial differential equation for variable ϕ or a linearized form of the advection operator for ϕ. The residual can also be based on the fine-scale turbulent kinetic energy partial differential equation. This kinetic energy-based residual can also be interpreted as a LES subgrid scale model [7]. Please refer to the Nalu theory manual for more information on this developing approach.

4. PRODUCTION RESULTS

Several production cases have been run to determine the robustness of the hybrid CVFEM/DG scheme. In this section, only two of studies will be briefly outlined. The first demonstration case is a spinning cube while the second is a production Vestas V27 turbine.



(a) P1P1-R1.

(b) P2P2-R0.

Figure 8: Spinning cube at 0.05 seconds for the P=1 and P=2 order; volume-rendered Q-criterion.

4.1 Spinning Cube

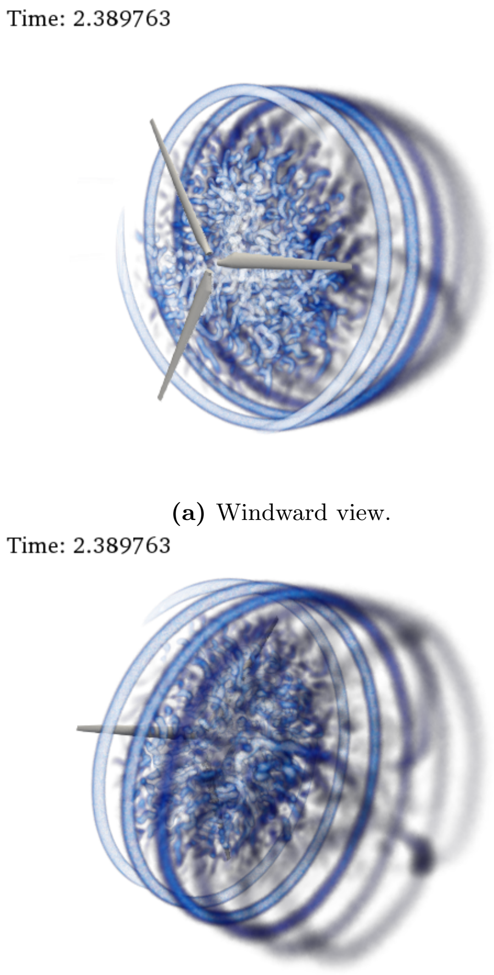
The spinning cube proof-of-concept study outlined in this report is an extension of Zhang [13], however, the Reynolds number has been increased to 4,000 and the RPM to 3600. In this study, the entropy-viscosity approach of Guermond is used both for stabilization and as a large-eddy simulation (LES) model. Mixed order interfaces are again used as with the viscous vortex problem. Figure 8 outlines the volume-rendered Q-criterion at 0.05 seconds for the refined P=1 and base P=2 mesh. Due to space, the full set of mesh refinements are not included in this report. Drag coefficient predictions for the two P=1 and P=2 simulations are 1.1565 and 1.1576, respectively. For non-spinning cubes, theory suggests a value of unity. Although simulations shown for the P=1 and P=2 mesh are using the same number of elements, the increased turbulent flow structure provided in the volume-rendered Q-criterion is noteable.

4.2 V27bo; Blades Only

In this section, a V27bo set of airfoils are rotated at a RPM of 5.0725. This mesh study does not include the tower and nacelle as it represents the first production simulation run carried out in FY17. This effort was to simply test the outlined DG scheme at a production V27 scale (26 meter diameter rotor without tilt). Based on the inlet velocity of 10 $\frac{m}{s}$, the tip speed ratio is approximately 6.8. The baseline hybrid mesh is comprised of approximately 150 million elements consisting of low-order hexahedral, tetrahedral, pyramid, and wedge topologies. At the DG interface, the hexahedral topology is used. The max cell Peclet number is approximately seven million. The Wall Adapting Local Viscosity (WALE) LES model is activated. Figure 9 outlines the volume-rendered Q-criterion from the windward and leeward perspective.

4.3 V27; Full Turbine

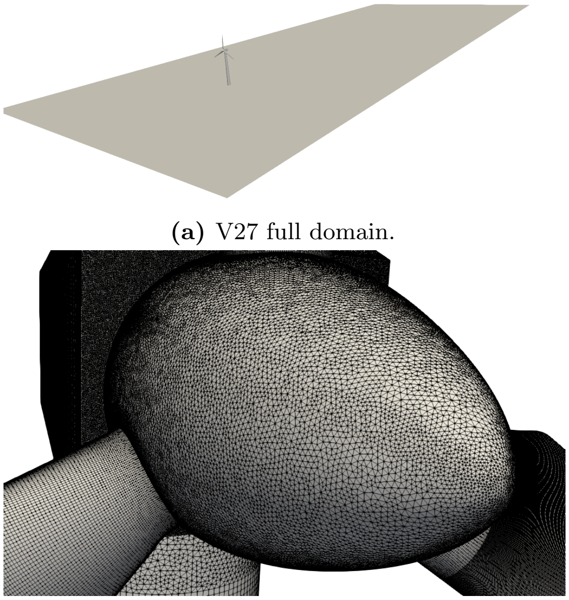
In this section, a full production mesh of the entire V27 turbine is provided. Tower and nacelle geometry is now included. In this simulation, design conditions are respected (RPM of 43) with a cross flow of 7.6 $\frac{m}{s}$. Again, the WALE model is used. The DG interface now includes the tetrahedral topology, which will be seen in later sections, required additional algorithmic improvements. Figure 10 provides a of geometric description of the domain. Again, the mesh is hybrid in nature using the full set of topologies supported in



(a) Windward view.

(b) Leeward view.

Figure 9: Vestas V27bo rotor simulation at 2.389763 seconds; volume rendered Q-criterion.



(a) V27 full domain.

(b) Close-up of the cone section and nacelle; surface mesh lines are in black.

Figure 10: Vestas V27 mesh attributes for the full turbine blade.

the code base. The mesh is using approximately 160 million elements. Figure 11 is a very preliminary velocity field prediction at a cut plane with the matching 4-degree wing tilt (shown from the windward and leeward perspective). The nonlinear stabilization operator (NSO) is in use as is a second-order accurate temporal integration scheme (BDF2).

Finally, the newly developed Kokkos-based interior elemental algorithmic kernels, which were outlined in the FY17/Q4 ExaWind milestone, have been activated for this hybrid full-scale V27 mesh. As noted in the previous FY17/Q4 milestone, the transition of the `MasterElement` implementations to the Kokkos-shared memory view data structures facilitated this effort as did other core algorithmic `Kernel` development. In the current simulation shown in Figure 11, only SIMD is active with the number of threads set to unity.

5. SOLVER AND APPLICATION CODE BENCHMARKING

In this section, we present strong-scaling results for the V27 moving-mesh simulations with several refinement levels. These are the meshes, V27 41 (refinements R0 66M and R1 500M elements), and V27 41a (refinements R0 166M and R1 1.3 Billion elements; a R2 mesh, which was greater that four Billion elements, was generated, however, has not yet been tested). The V27 problem presents several difficult challenges for the pressure continuity solver. The mesh is hybrid as described above containing hexahedral, tetrahedral, pyramid, and wedge finite elements. The element aspect ratios can be large near surfaces. Further, the mesh is non-conforming and consists of different blocks or regions with mesh movement. Consequently, the momentum and continuity linear-system matrices must be reinitialized and reassembled each time step. These simulations were run in first-order accurate in time mode, but second-order accurate results should have very similar times. In addition, the associated preconditioners are reconstructed for these matrices. The set-up time for these operations can be a significant overhead that may not scale well and the CPU time has been tracked with timers in the Nalu code. Finally, the number of nodes in the mesh defines the matrix row size. Hybrid mesh element counts are typically a factor of six larger than the number of nodes, especially when the majority of elements are tetrahedral in nature.

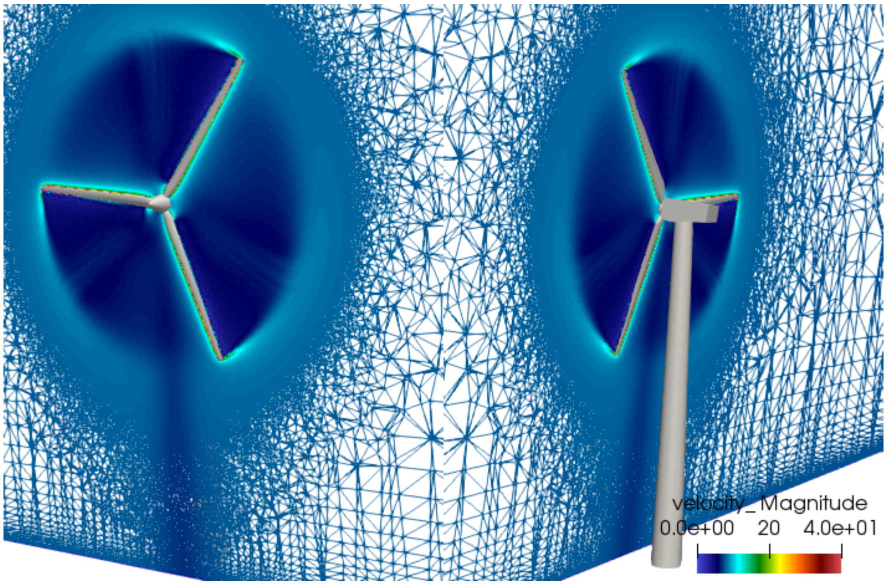


Figure 11: Preliminary V27 full tower results, shown velocity shadings from the windward and leeward perspectives; simulations are ongoing and seemingly stable.

5.1 Solver Descriptions

In the description of numerical methods employed in the Nalu model, it was noted that the pressure continuity equation solver is based on a preconditioned non-symmetric GMRES Krylov iteration. The preconditioner consists of one V-cycle of an algebraic multigrid (AMG) method. Initially, the Nalu continuity solver was interfaced to the Trilinos MueLu SA-AMG framework. MueLu implements a smoothed-aggregation coarsening algorithm to construct the coarse matrices in the V-cycle. Unsmoothed aggregation or prolongation is also possible and has proven to be more effective for the V27 problem as discussed below. More recently, Nalu has been interfaced to the Hypre solver framework from the CASC group at LLNL. Hypre provides the GMRES iterative solver and boomerAMG preconditioner. The classical Ruge–Stüben C-AMG coarsening algorithm is implemented by boomerAMG. SA-AMG and C-AMG represent the two fundamental algebraic-multigrid (AMG) approaches to creating preconditioners. They will be compared here on the basis of set-up times, solve times and overall simulation wall-clock times in a strong-scaling study.

The GMRES Krylov iterative solvers employed by Trilinos and Hypre differ in several respects that can affect the parallel performance and scaling. The main difference lies in the Gram-Schmidt orthogonalization algorithms for the Krylov vectors and the communication associated with global reductions.

1. Hypre MGS-GMRES iterative solver: Modified Gram-Schmidt; numerically Krylov vectors loss of orthogonality but more stable; requires separate inner products for Krylov basis; less scalable.

2. Trilinos-Belos ICGS-GMRES iterative solver: Iterated classical Gram-Schmidt; requires two passes of Gram-Schmidt(re-orthogonalization) to recover stability of MGS; only one large vector `MPI_Allreduce` required for inner products; highly scalable.

The MGS-GMRES is widely implemented but may be less scalable at large processor counts. A smaller Krylov subspace dimension or restart will mitigate these affects. The iterated ICGS-GMRES trades computation for communication and has proven to be more scalable.

Given the linear system of equations $Ax = b$, smoothed aggregation is based on strongly connected neighborhoods:

$$N_i(\varepsilon) = \left\{ j : |a_{ij}| \geq \varepsilon \sqrt{a_{ii}\, a_{jj}} \right\}$$

and the prolongation operator P is constructed by defining C_i as the set of nodes in aggregate i and $\tilde{P}_{ij} = 1$,

$i \in C_j$, and 0 otherwise. The filtered matrix $A^F = (a_{ij})^F$ is defined as $a^F_{ij} = a_{ij}$, $j \in N_i(\varepsilon)$ and 0 otherwise.

$$a^F_{ij} = a_{ii} - \sum_{j=1}^{n} \left(a_{ij} - a^F_{ij} \right)$$

The prolongation operator is the damped Jacobi smoothing $P = \left(I - \omega \, D^{-1} \, A^F \right) \tilde{P}$ where $D = \mathrm{diag}(A)$, A_F is the filtered matrix, and $\omega = 3/(\, 4 \, \lambda_{\max}(\, D^{-1} \, A) \,)$. Unsmoothed prolongation is obtained by setting $\omega = 0$. This has proven to be an effective coarsening strategy for the V27 problem due to the lower sparsity of the tentative prolongator $\tilde{P}$ and the resulting coarse matrices $A_c = \tilde{P}^T \, A \, \tilde{P}$.

In the classical Ruge-Stüben coarsening algorithm two nodes u_i and u_j are said to be connected if $a_{ij} \neq 0$. Given a threshold value $0 < \theta \leq 1$, the node (point) u_i is strongly connected to u_j if

$$|a_{ij}| \geq \theta \max_{k \neq i} |a_{ik}|$$

The set of grid points selected to be part of the coarse grid is called C. An element in C will be called a C-point and these are chosen based on the number of strong connections to each node. The remaining points left out of C are called fine points and belong to the set F. Points in F are called F-points. The prolongation operator is constructed row wise using interpolation. If i is a coarse node, then the i–th row is the identity. If i is a fine node, then a row of a weight matrix is employed. Thus, row-wise coarsening tends to emulate to geometric approaches.

The pressure continuity solver contribution to the overall simulation wall-clock time is determined by several factors. The number of GMRES solver iterations required to achieve a given tolerance level may be small. However, the smoothing sweeps applied to the coarse matrices in the V-cycle can be expensive if these matrices are not sufficiently sparse. The sparsity (or complexity) of these matrices is determined by the number of non-zeros per row and coarsening rates. SA-AMG generally coarsens by 3×, whereas C-AMG coarsens by 2×. The C-AMG interpolation stencil width changes the complexity and aggressive coarsening can change the rate to 4×. Unsmoothed prolongation in SA-AMG with $\omega = 0$ can further improve sparsity. The number of V–cycle levels and size of the coarse system also affect the GMRES convergence rate, amount of parallel communication and run-time.

5.2 Solver Performance

For the purposes of our V27 strong scaling study, the optimal algorithmic and parameter choices for MueLu were determined by J. Hu [9] as follows. Unsmoothed prolongation $\omega = 0$ was employed with MueLu, leading to sparser coarse matrices. A distance Laplacian dropping scheme is applied with a 0.02 threshold drop tolerance. An implicit prolongation operator is retained. Re-balancing is also applied to the coarse matrices to improve parallel performance. Four sweeps of an L_1 Gauss-Seidel smoother are performed at each V-cycle level (both up and down the hierarchy) and the coarse level solver is SuperLU. For the V27 problem, the V-cycle hierarchy typically contains eight levels.

Coarsening for the Hypre C-AMG is based on the parallel modified independent set (PMIS) algorithm of Hans de Sterck [2] allowing for a parallel set-up phase. A transposed prolongation operator is retained for triple-matrix RAP products. Aggressive coarsening is applied on the first two V-cycle levels with multi-pass interpolation and a stencil width of two elements per row. The remaining levels employ extended+i interpolation. Interpolation truncation level 0.25 is specified together with a maximum interpolation stencil width of two matrix elements per row. The smoother is a hybrid Gauss-Seidel relaxation scheme. The coarsening rate for the V27 problem is roughly 7× with eight levels in the V-cycle for Hypre. Operator complexity as defined by Hypre is close to 1.1 indicating more efficient V-cycles with aggressive coarsening, however, an increased number of GMRES iterations are required compared to standard coarsening.

Strong scaling studies were conducted using two supercomputers, Peregrine at NREL and Cori at NERSC. Peregrine nodes contain Xeon E5-2670 v3 Haswell processors, 2×12 core sockets, 64 GB DDR4 memory, inter-connected with an Infiniband network. Cori nodes consist of E5-2698 v3 Haswell processors clocked at 2.3 GHz, with 2×16 core sockets, and 128 GB DDR4 memory. The inter-connect is the Cray Aries. Although the model can support threads with kokkos and OpenMP, for the purposes of these runs, one MPI rank per core was employed. The Nalu+Hypre configuration of the model is compared with the Nalu+MueLu version. The momentum solver in all cases is based on the Trilinos-Tpetra solver stack. The Belos GMRES momentum solver is preconditioned with a symmetric Gauss-Seidel relaxation.

5.2.1 V27 41 Mesh

The unrefined R0 mesh for the V27 41 problem consists of 66 Million elements. One step of uniform refinement using the “percept” tool from Sandia results in the R1 mesh consisting of 500 Million finite elements. Initially, the V27 41 R0 case was run on Peregrine using 1536 cores for 40 time steps. The resulting number of elements per core is close to 42K. The total simulation time for this run was 487 sec for Nalu+Hypre and 507 sec for Nalu+MueLu. The solver component times and average iteration counts for the momentum and pressure continuity solvers are provided in Table 1 below. We note that the ‘init‘ time for the Nalu+MueLu continuity solver is 10× the ‘init’ cost in Nalu+Hypre. The solve times are roughly equivalent, however, the iteration count for the Hypre solver is 5 versus 10 for MueLu. Thus, the cost per Hypre iteration is larger. This can be partially attributed to GMRES communication overheads and the sparsity of the coarse matrices.

Component		% Time		% Time
myLowMach Solver				
misc	4s	0.1	3s	0.1
Momentum Solver	Belos/SGS		Belos/SGS	
init	76.2s	16	74.8s	17.0
assemble	71.1s	15	71.0s	14.0
load complete	57.1s	12	58.6s	7.0
solve	43.0s	8	43.6s	8.5
precond setup	0.4s	0.0	0.4s	0.0
misc	9.7s	0.2	9.1s	0.2
linear iterations	5		10	
Continuity Solver	Hypre		MueLu	
init	3.9s	0.1	30.9s	6.1
assemble	11.4s	2	12.6s	2.0
load complete	15.9s	3	8.6s	1.6
solve	81.9s	17	108.6s	21.3
precond setup	30.9s	6	28.3s	5.5
misc	14.7s	3	14.3s	2.8
linear iterations	17		22	
IO Time	5.5s	0.1	8.0s	0.2
Non-conformal BC	45.0s	2	20.9s	4.0
Skin Mesh	0.1s	0	0.1s	0.0
Other	4s	0.1	15.0s	0.1
total time	488	100	507s	100

Table 1: Total simulation time with solver component times for V27 41 R0 66M element mesh showing strong scaling performance up to 1536 cores on Peregrine. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu.

The V27 41 R1 mesh simulation was run on up to 12K cores of the NERSC Cori supercomputer for 100 time steps. The total R1 simulation time on 12K cores using Nalu+MueLu is 3413 seconds and for Nalu+Hypre the time is 2923 seconds. The simulation times at different core counts are plotted in Figure 12. Parallel efficiency is plotted in Figure 13. The solver component times and average iteration counts for the momentum and pressure continuity solvers are provided in Table 2 below. The simulation times are also provided in a separate Table 3 below with the number of DOF’s or FE nodes per core.

The momentum equations initialization is the largest time consumer and requires 25% of the simulation time for Nalu+Hypre. We note that the initialization time for Nalu+MueLu continuity represents 18% of the simulation time. The initialization time “init” is related to the Trilinos-Tpetra parallel communication algorithms and related graph construction. The AMG “precond setup” time is associated with construction of the V-cycle hierarchy, including coarsening, smoother matrix splitting, re-balancing and communication. Both the “init” time and “precond set-up” time increase as a percentage of run time as the number of cores increases to 12K, thereby, inhibiting scaling. The “init” time for the Nalu+Hypre simulations is 60× less than for Nalu+MueLu and accounts for much of the discrepancy in total simulation times.

Component		% Time		% Time
myLowMach Solver				
misc	6.9s	0.9	7.2s	0.2
Momentum Solver	Belos/SGS		Belos/SGS	
init	721.1s	25.0	696.0s	20.0
assemble	189.9s	6.5	191.0s	5.6
load complete	227.5s	7.7	222.5s	6.5
solve	413.3s	14.0	352.5s	10.3
precond setup	1.0s	0.03	1.1s	0.03
misc	24.7s	0.9	23.9s	0.7
linear iterations	11		10	
Continuity Solver	Hypre		MueLu	
init	10.3s	0.3	635.0s	18.5
assemble	36.2s	1.2	33.8s	1.0
load complete	59.7s	2.0	36.5s	1.0
solve	502.7s	17.2	580.7s	17.0
precond setup	287.9s	9.8	204.1s	6.0
misc	32.3s	1.1	43.5s	1.3
linear iterations	12		35	
IO Time	24.6s	0.8	16.0s	0.5
Non-conformal BC	303.2s	10.4	305.1s	8.9
Skin Mesh	8.4s	0.3	3.6s	0.1
Other	73.3s	2.5	59.4s	1.7
total time	2923s	100	3413s	100

Table 2: Total simulation time with solver component times for V27 41 R1 500M element mesh showing strong scaling performance up to 12288 cores on Cori. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu.

The “init” time for the Muelu column under the “Continuity Solver” section, involves our initialization and filling of the Tpetra CrsGraph object which holds the sparsity structure of the matrix for each of the linear systems. We are currently undertaking an effort to reduce the “init” time for Tpetra by providing it with sharp initial bounds for the length of each row. This will allow Tpetra to avoid dynamic allocations and re-allocations during the filling stage. We will also be examining ways to reduce the amount of parallel communication that Tpetra must perform to determine internal structures, etc.

Cores	Nodes/Core	Nalu+Hypre (s)	Nalu+MueLu (s)
2048	111K	7873	9920
4096	55K	4568	5710
8192	27K	3186	3808
12288	18K	2923	3413

Table 3: Total simulation times for 100 time steps of the V27 41 R1 500M element mesh showing strong scaling performance up to 12k cores. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu

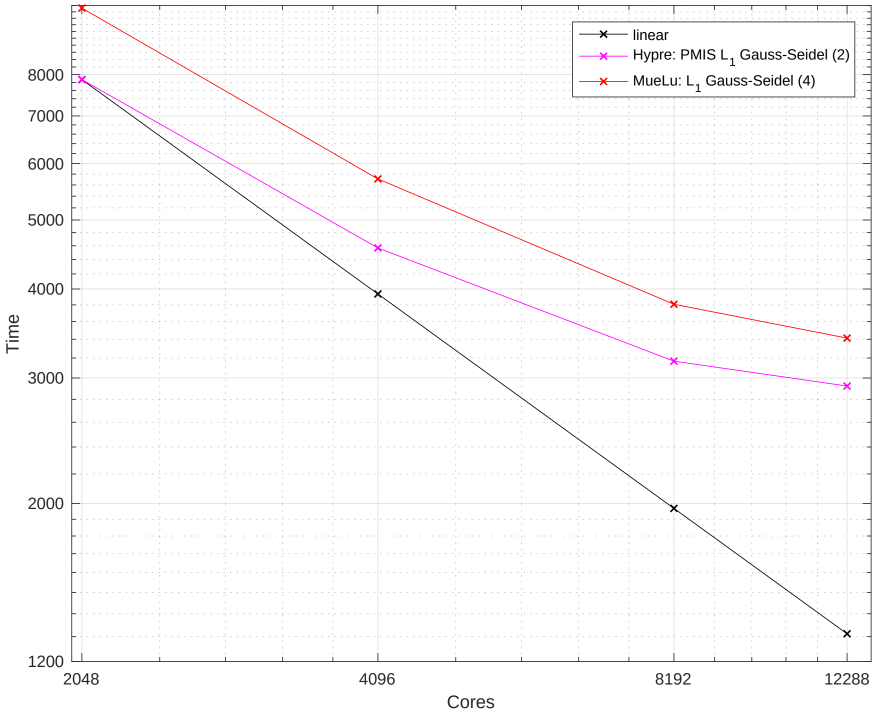


Figure 12: Total simulation times for 100 time steps of the V27 41 R1 500M element mesh showing strong scaling performance up to 12k cores. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu

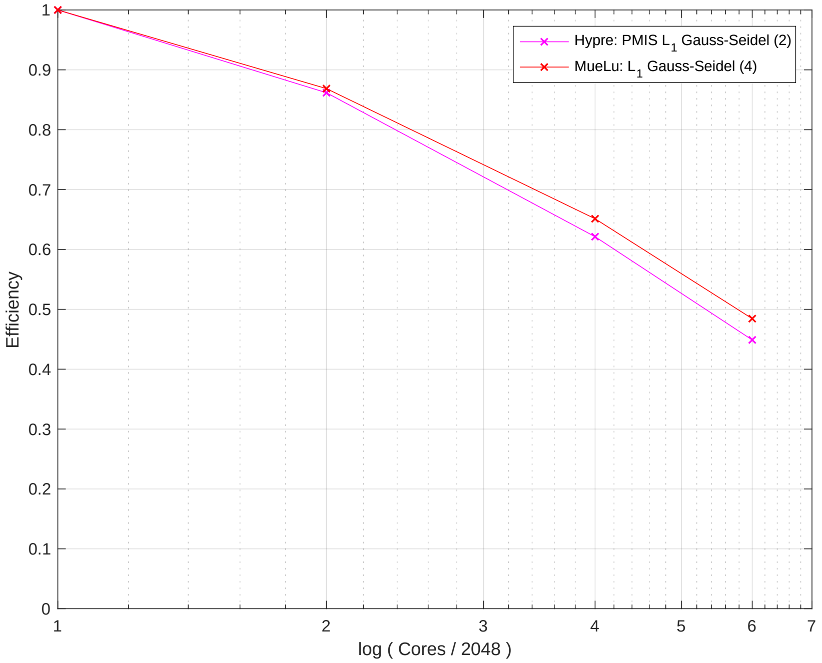


Figure 13: Parallel efficiency plot for V27 41 R1 500M element mesh showing strong scaling performance up to 12Kcores on Cori. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu.

5.2.2 V27 41a Mesh

The unrefined R0 mesh for the V27 41a problem consists of 166 Million elements. One step of uniform refinement using the "Percept" tool from Sandia results in the V27 41a R1 mesh consisting of 1.3 Billion finite elements. The V27 41a R0 mesh simulation was performed on 3072 cores on Peregrine for 100 time steps. The resulting number of elements per core is 54K. The total simulation time for this run was 1149 sec for Nalu+Hypre and 1249 sec for Nalu+MueLu. The solver component times and average iteration counts for the momentum and pressure continuity solvers are provided in Table 4 below. Here we note that the "init" time for the continuity solver is 9× larger for Nalu+MueLu. The solve cost is also larger, however, the 'precond setup' is slightly lower . The momentum "init" is still the dominant cost in the simulations.

Component	% Time		% Time	
myLowMach Solver				
misc	8s	1.0	8s	1.0
Momentum Solver	Belos/SGS		Belos/SGS	
init	198.9s	17	197.2s	16.0
assemble	118.7s	10	118.9s	9.5
load complete	165.1s	14	164.3s	13.2
solve	104.0s	9	102.s	8.3
precond setup	0.8s	0.0	0.8s	0.0
misc	25.2s	2.2	24.0s	4.4
linear iterations	5		10	
Continuity Solver	Hypre		MueLu	
init	10.0s	1.0	92.7s	6.1
assemble	25.3s	2	23.1s	2.0
load complete	52.2s	4	30.2s	2.5
solve	217.9s	18	266.3s	21.3
precond setup	100.3s	9	73.9s	6.0
misc	41.8s	3	41.3s	2.8
linear iterations	16		21	
IO Time	10.5s	0.1	8.0s	0.1
Non-conformal BC	47.5s	4	48.6s	4.0
Skin Mesh	0.1s	0	0.1s	0.0
Other	37.0s	3.0	46.0s	4.0
total time	1149	100	1249	100

Table 4: Total simulation time with solver component times for V27 41a R0 166M element mesh showing strong scaling performance up to 3072 cores on Peregrine. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu.

The V27 41a R1 case with 1.3 Billion elements was simulated on 12288 cores on Cori for 20 time steps. The resulting number of elements per core is 105K. The total simulation time for this run was 948 sec for Nalu+Hypre and 1418 sec for Nalu+MueLu. The solver component times and average iteration counts for the momentum and pressure continuity solvers are provided in Table 5 below. In particular, note that the "init" time for the continuity solver is now 40× larger for Nalu+MueLu. The solve cost for MueLu has increased. The 'precond setup' for Hypre is slightly lower . The continuity 'solve' is the dominant cost along with the MueLu solve. Overall for the Nalu+MueLu run the total momentum and continuity 'init' time is 33% of the simulation time. An important observation is that the momentum equations solve time has increased together with the number of GMRES iterations and represents a larger percentage than the continuity pressure solver. The momentum solve may now require a multi-level AMG preconditioner. In all the results reported here, the SIMD improvements from the FY17Q4 milestone were included.

Component		% Time		% Time
myLowMach Solver				
misc	2.6s	0.3	2.6s	0.2
Momentum Solver	Belos/SGS		Belos/SGS	
init	256.7s	27	237.5s	16.7
assemble	35.6s	4	35.6s	2.5
load complete	65.8s	7	65.4s	4.6
solve	166.0s	17.5	240.3s	17
precond setup	0.5s	0.0	0.5s	0.0
misc	9.7s	1.0	9.8s	1.0
linear iterations	11		17	
Continuity Solver	Hypre		MueLu	
init	5.0s	1.0	199.3s	14.1
assemble	10.5s	2	8.2s	1.0
load complete	18.9s	2	11.6s	1.0
solve	122.6s	13	292.3s	20.6
precond setup	39.2s	4	62.2s	2.5
misc	14.7s	1.5	14.7s	1.0
linear iterations	20		38	
IO Time	36.1s	3.7	44.5s	3.1
Non-conformal BC	44.1s	4.6	46.7s	3.3
Skin Mesh	14.7s	1.5	22.7s	1.6
Other	70.0s	7.4	81.0s	6.0
total time	948	100	1418	100

Table 5: Total simulation time with solver component times for V27 41a R1 1.3B element mesh showing strong scaling performance up to 12288 cores on Cori. Shown are results for Belos/SGS+Hypre and Belos/SGS+MueLu.

5.3 Search and Custom Ghosting

The hybrid nonconformal scheme requires parallel searches to identify the opposing element from which opposing numerical fluxes are computed. This search is performed each time step when moving mesh simulations are active. The open-source code base currently supports two parallel searches: 1) an RTREE-based search provided by Boost and 2) a KDTREE-based search provided by the Sierra Toolkit (STK). As part of the Advanced Technology Development and Mitigation (ATDM) program, a new NGP-based parallel search is under development and has been deployed to an internal Sierra application code. This parallel search, which is based on the MORTON[3] patent, supports threading. At present, the open source Nalu interface is not activating this methodology as the above search is limited to an internal Sandia product. However, evaluation of this search is a planned activity for FY18. Parallel ghosting of opposing elements that are off-processor is managed by the STK `Ghosting` interface. This interface, which has been deployed in the production code base, allows a simple ability to provide the dynamic list of required ghosted entities for the scheme, i.e., opposing faces, opposing elements, and the nodes connected to each.

The benchmark for the current parallel search and custom ghosting was accomplished through the set of production V27 meshes. Results indicated that the ghosting and parallel search algorithmic component of the overall simulation time was always under 5% of overall simulation time. In a separate study, the performance disparity between KDTREE and RTREE was small. In general, the surface-based searches and modest custom ghosting events appear to be a low-consumer of the overall simulation time at this given computational scale.

As computational resources increase and the mesh sizes move beyond the 32-bit threshold, the nonconformal algorithm search and custom ghosting timings will be revisited. The cost of the parallel search can increase

[3] https://www.google.com/patents/US9396512

by an errant user specification of search tolerance. With such events, too many candidate opposing elements are provided to the fine search from from the coarse search. This increases the custom ghosting size and also requires more elements to query to identify the "best" opposing set of element. Efforts to possibly incorporate intelligent search tolerances are expected later in the year. At present, several modifications to the former nonconformal algorithm were completed in FY18/Q1, most notably a "dynamic" algorithm in which the custom ghosting object is not destroyed and recreated (see Section 6).

5.4 Hybrid Topology Kokkos-Kernels

The FY17/Q4 effort drove the development of the Nalu `Kernel` that provides a computational algorithm construct using Kokkos data structures. These kernels are integrated within a generalized STK/Kokkos `SolverAlgorithm` design that allows for a team-based nested Kokkos-parallel functionality. Moreover, the extensive usage of the STK `SIMD` interface allows for the interleaving of data to remove the requirement to rely on the compiler for vectorization. At the end of FY17/Q4, this methodology had been prototyped and demonstrated for homogeneous hexahedral topology code path. In FY18/Q1, the full transition to Kokkos was acomplished for all interior topology matrix contributions of immediate interest to the wind application space. The complete transition of the `MasterElement` core methods, e.g., gradient operators, area vector, etc., from FORTRAN-based implementations to shared memory Kokkos `Views` was accomplished.

The benchmarking of the Kokkos-based `Kernel` approach for the production V27 41a mesh (with tower and nacelle) indicated that overall simulation time was approximately 12% faster with the Kokkos-based Nalu `Kernel` design. In this study, 100 timesteps were used; the platform was the SNL institutional Ghost cluster (Intel Broadwell with Omni-Path high speed interconnect). The number of threads used was unity, however, the SIMD interface was active. Therefore, only the matrix assembly timings include possible speed-ups solely from the SIMD interface. In the current SIMD interface, data is unpacked prior to interfacing with the solver object. Assembly timing for key kernels such as the momentum advection/diffusion contribution noted a 2.3x speed-up (from 179 seconds to 78 seconds). For continuity assembly, which is only a simple Laplace operator, timings reduced from 23 seconds to 17 seconds. As noted, in this study flat MPI was used without threading. As such, the speed-ups noted here are solely due to the new homogeneous algorithm design in which a low level computational `Kernel` is templated on topology traits along with an underlying `SIMD` design. This study provides confidence that the new developer-friendly multi-dimensional array Kokkos/SIMD abstractions does not penalize performance.

6. APPLICATION IMPROVEMENTS

In this milestone effort, several application code improvements have been deployed. This section describes the set of notable code improvements and the associated technical drivers for the work.

6.1 Faceting

As the project continues to move down the path for higher-order discretizations, it is expected that the faceting at interfaces due to the previous low-order representation of a curved surface will improve. However, a primary finding in early full-scale V27 production simulation studies was that generalized unstructured tetrahedral element interfaces at the DG interface required more mesh resolution than structured, hexahedral interfaces[4]. In order to explore this sensitivity, a representative tetrahedral mesh was created that only included an inner rotating disk at 4-degree tilt in order to match the formal V27 size configuration, i.e., O(30)meter rotating inner cylinder. The flow configuration was set to inflow, open and symmetry to allow for a purely uniform flow. Varying mesh resolutions using either hexahedral or tetrahedral topologies were run. The revolution rate was set at a value more that five times the highest design configuration (30 radians/second) to push the algorithm and mesh velocities at the representative radius. Finally, mesh resolution was varied from 1 meter to 0.25 meters. In this particular configuration, any non-streamwise velocity component can be considered an L_∞ error.

[4] Here, full-scale is defined by the tower, nacelle, cone and blade geometry that drove tetrahedral topologies at the nonconformal interface

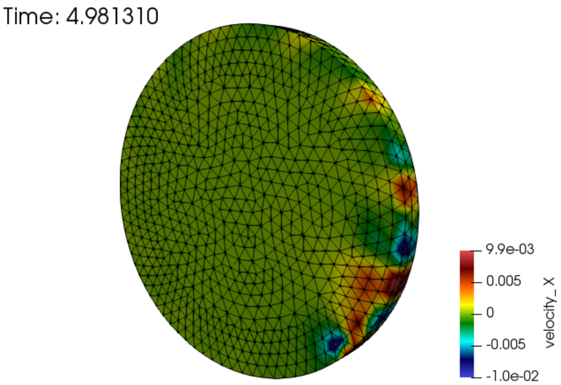


Figure 14: Velocity norm for baseline surface-based hybrid method.

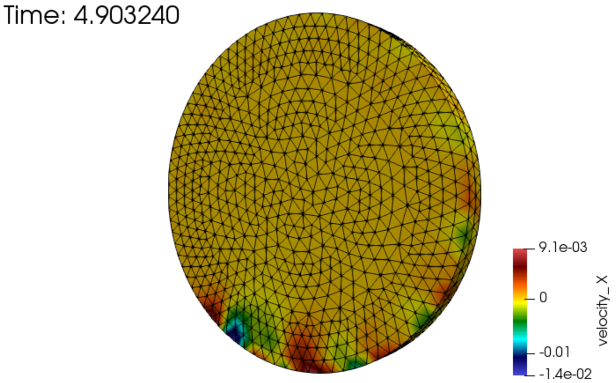


Figure 15: Velocity norm for the overlapping DG scheme in which the mesh blocks now have a ten percent overlap.

For all of the near-nonconformal interface meshes that used quasi-structured hexahedral meshes, the error norm due to the moving mesh interface was $\ll 0.01$ percent. Nonlinear norms remained in the O(1e-14) range indicating pure one-dimensional flow. However, for tetrahedral meshes, the norms were considerably higher. Figure 14 outlines the error in the x-component of velocity for the mesh resolution of one meter for the tetrahedra-based mesh; shown are the x-velocity component values at the inner rotating cylindrical mesh block.

In Figure 15, the surface-based approach has been changed in favor of an overlapping algorithm. In this formulation, the inner mesh overlaps the background stationary mesh. Therefore, the exposed surface integration points now are mapped to opposing elements. The standard hybrid algorithm applies, however, rather that an opposing surface being used to compute gradients, length scales, etc., the owning element does. This overlapping scheme has similar attributes to an overset scheme, however, does not include deactivation of overlapped elements. Moreover, if the overlap is too large, then the first layer of opposing elements can be missed. In such cases, the search is also more complex in that the set of elements with an exposed face may not be the full set of best candidate elements. As seen from the figure, the improvement in error norm is not appreciable and suggests another underlying issue. Nevertheless, this formulation can provide a unification of the sliding mesh and overset algorithm. Currently, the overset algorithm is best described as an implicit constraint-based approach, however, could easily be cast within a weak surface integration as noted in this section.

The above set of findings from the numerical experiments suggest that there may be an algorithmic nuance or bug within surface integration of the mesh velocity as a non-rotating inner disk does not exhibit any appreciable numerical error. Specifically, the advection term for any scalar ϕ can be written as,

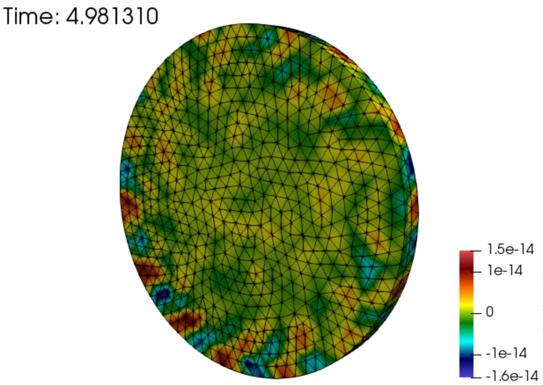


Figure 16: Velocity norm for volumetric mesh velocity source term approach.

$$\int_{\Gamma} \rho \left(u_j - v_j\right) \phi n_j d\Gamma, \tag{12}$$

where v_j is the mesh velocity and $u_j - v_j$ is the velocity relative to the mesh. In this usage of the Gauss-Divergence theorem, which is common in finite volume schemes, volume integrals have been transformed to surface contributions. In the case of general tetrahedral meshes, nodes at the curved surfaces can be out of problematic and may provide additional error. An alternative to the above advection form is as follows:

$$\int_{\Gamma} \rho u_j \phi n_j d\Gamma - \int_{\Omega} \frac{\partial \rho \phi v_j}{\partial x_j} d\Omega. \tag{13}$$

In the above equation, the volumetric gradient operator can simply be computed using elemental CVFEM shape function derivatives. Figure 15 provides the error norms for the one-meter resolution tetrahedral mesh. As shown by the results, the error is now commensurate with the hexahedral mesh results; O(1e-14).

In practice, surface and volume integration should produce numerically the same result even in the presence of complex boundary data at curved surfaces. To further understand each methodology, the divergence of mesh velocity was post processed using either surface of volume integration. The L_2 lumped projected nodal gradient for the mesh velocity, using surface integration, is given by,

$$G_j v_i = \frac{\int v_i n_j d\Gamma}{\int d\Omega}, \tag{14}$$

while for volume integration, it is as follows:

$$G_j v_i = \frac{\int \frac{\partial v_i}{\partial x_j} d\Omega}{\int d\Omega}. \tag{15}$$

For simple solid rotation, the divergence of the mesh velocity should be zero.

Figure 17 outlines the L_∞ norm for the divergence of mesh velocity for the simple rotating tetrahedral mesh. Although the mesh velocity divergence is zero for both schemes at the interior nodes (not shown in these images), at the surface location, the surface-based reconstruction is clearly non-zero. However, as suggested from the full fluids experiments, the volumetric-based approach is nearly machine precision zero. In this experiment, mesh velocity is provided based on the analytical solid rotation and not based on numerical time derivatives of the mesh displacement. These findings clearly suggested a bug in the underlying tetrahedral/triangle topology in the Nalu code base.

Further investigations to identify the source of this nonintuitive numerical finding were carried out in both the Nalu and Sierra Thermal/Fluids code base Fuego, which has adopted the hybrid nonconformal methodology for external non-wind applications. A full suite of unit tests for all volume and surface topologies pointed to a conceptual issue with quadrature locations for the `Tet4SCS`, `Tri3DSCS`, and `Tri2DSCV` elements. After changing quadrature points for these topologies, Figure 18 outlines the expected numerical norm using

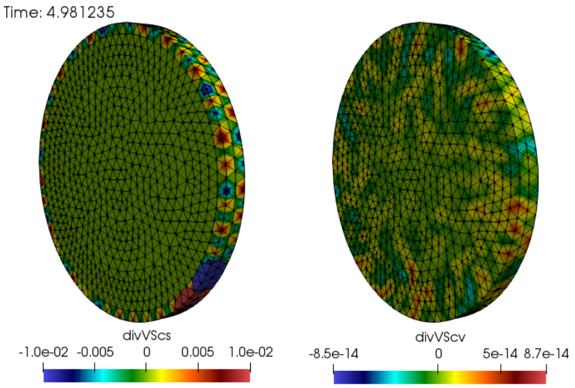


Figure 17: Post processed mesh velocity divergence comparing surface- and volume-based integration.

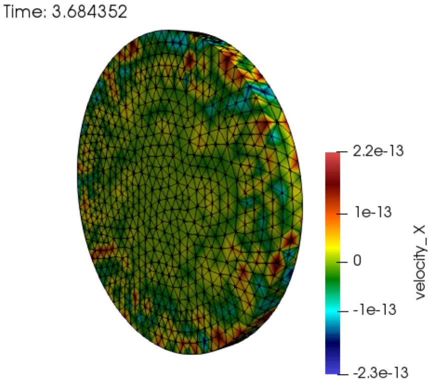


Figure 18: Velocity norm for surface-based mesh velocity approach when using the newly defined tetrahedral quadrature points.

the surface-based quadrature approach. Currently, both subcontrol volume surface and volume approaches for the tetrahedral topology provide like results. As the `Tri3DSCS` topology exists in the `Wedge6` and `Pyramid5`, further verification of all CVFEM supported topologies will be carried out in the FY18/Q2 ExaWind project time frame. A shared story between the ExaWind and the Sierra Thermal/Fluids team will be executed in FY18/Q2 that will highlight the synergy between the ASC and ECP portfolio. At the time of this report, all topologies (not including pyramids) have been converted.

6.1.1 Opposing Normal Treatment

In the case of highly faceted low-order representation of a curved surfaces, the current and opposing normal definition can be misaligned. In such cases, a trivial change that has significantly improved simulation quality, most notably in the pressure field, is to define the opposing normal as,

$$n_j^o = -n_j^c. \tag{16}$$

In the above form, the current exposed face normal definition prevails over the actual opposing normal based on geometry.

6.2 Matrix Reduction Techniques

In general, an implicit matrix contribution for the complete non-conformal interface discretization stencil is desired. As the core use-case in wind energy requires rotation, a dynamic stencil matrix assembly and solve interface is required. In the current production implementation, a parallel search, parallel ghosting (using the STK custom ghosting interface) and matrix reinitialization are performed at each time step. Several possibilities to remove the need and/or frequency of the matrix reinitialization exists. For example, the opposing column entries in the Jacobian can be removed and repaired by iteration. Second, matrix entries can be lagged over the simulation. Finally, the matrix stencil can be "bloated" to include elements that will be part of the stencil at future simulation times. All methods are expected to provide a trade-off between stability and performance.

6.2.1 Removal of Matrix Entries for Opposing Elements

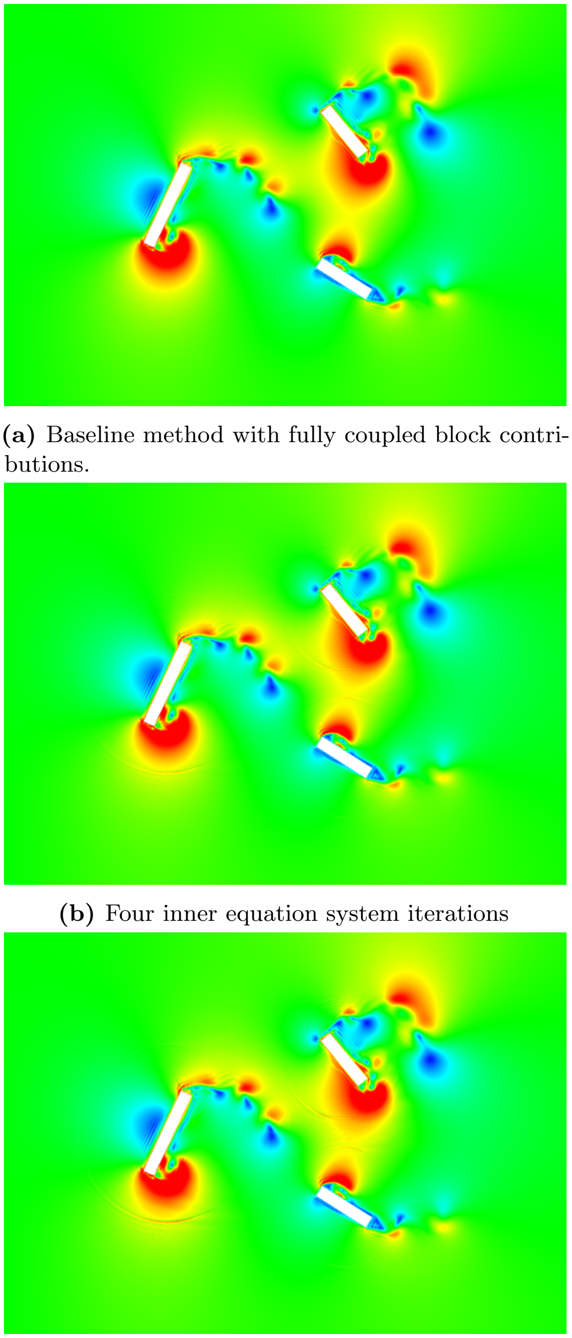
Omitted matrix entries for the opposing elements can be repaired by inner iteration. This approach allows for a constant matrix stencil at the expense of missing information in the Jacobian. In this high-level study, inner iteration counts of four and two are compared against the baseline algorithm, i.e., full matrix entries. The canonical test case is that of three independently rotating blades in a two-dimensional configuration. Figure 19 outlines a typical example comparing the baseline method with an approach that removes the opposing matrix entries, however, adds a set of inner iterations over the given PDE of two or four. When no inner iterations are performed, the simulations diverge. This stability issue is due to the elliptic pressure Poisson system being decoupled between the stationary and moving blocks. The resulting predicted pressure field is used within the velocity projection step. Given a corrupt pressure projection step, the velocity is contaminated and stability issues are found. Using four inner iterations results in a stable simulation with only slight distortion of the flow field. The usage of two inner iterations is stable, but results in appreciable interface noise. However, the increased cost of the overall simulation when using four iterations, i.e., the minimum number found in this study to produce a satisfactory interface velocity field, simulation time increases more than a factor of two. This is due to the simple fact that the Poisson solve required for the continuity enforcement represents an appreciable cost. Therefore, reducing Jacobian entries to avoid solver reinitialization events is accompanied by higher solver times. It was determined that this path forward was not satisfactory due to the increase in simulation cost required to adequately propagate information for the elliptic continuity system.

6.2.2 Lagging Matrix Entries

A second approach explored to reduce matrix reinitialization events is to lag (in time) the matrix entries. In other words, the matrix at any given time may include connectivity over the full moving and stationary blocks, however, can be out of date. The code was modified to selectively reuse the linear system and preconditioner. The same three-blade simulation case was used to evaluate this approach. Findings for this method were mixed. Specifically, for stationary meshes, the lagging of matrix entries was robust even at high Courant numbers. However, with inner mesh revolution, a substantial Courant stability limit was noted (less than approximately two) when the frequency of reinitialization events was greater than approximately four. Again, a tradeoff between stability and performance was noted. This methodology, which was developed on a fork of Nalu, was not promoted to the production code base.

6.2.3 Increase in Mesh Stencil

The final methodology under construction is the ability to increase the parallel search tolerance or percentage of bounding box expansion such that the list of possible opposing elements is larger than simply the true closest opposing element. Therefore, the coarse search identifies a set of opposing elements that forms the "bloated" stencil while the fine search determines the best opposing element. The non-active columns in the coarse search are provided zero entry values. The underlying nonconformal data structure retains knowledge over the "active" stencil and "available stencil" by storing global face/element IDs of the opposing element set. When the mesh moves, the increased search tolerance returns a set of elements from the coarse search. If the active "best" opposing element is found in the old set of face/element IDs, the matrix graph is re-used and setup is delayed. In this formulation, no loss of matrix information is noted. However, there exists both an increased stencil size in the linear system and the custom ghosting object is required to now have state, e.g., the current set of ghosted face/elements from the latest search and the previous set of ghosted face/elements. In general, the custom ghosting object is also larger due to the increase search tolerance.



(a) Baseline method with fully coupled block contributions.

(b) Four inner equation system iterations

(c) Two inner equation system iterations

Figure 19: Velocity flow field shadings outlining the effect of omitting opposing matrix entries in favor of inner iterations.

In a typical sliding mesh wind energy application, which may be run at a maximum Courant number of ten, the simulation time step can be O(5e-5) due to the high velocities near the tip of the wing. However, for a diameter of approximately thirty meters, the Courant number will be far less than unity. Essentially, the mesh stencil at the nonconformal inteface is varying very slowly over the time scale of the fluid motion. This allows for a large savings in setup costs. For the three-blade example outlined in Figure 19, preliminary results of this bloated stencil suggest that over a fifty time step interval, re-initialization costs can be reduced from fifty to six reinitialization events. If the search tolerance is doubled, the resulting initialization count over the fifty time steps is further reduced to three events. Efforts to now interface the reuse ability to the linear system construction are underway and expected to be completed early in FY18/Q2. At that point, the increased cost of matrix assembly, which is due to the increased stencil, will be evaluated.

6.3 Search and Custom Ghosting

Opposing elements that exist on the opposing block boundary are identified by a geometric proximity search, and then brought to the current processor by using a 'custom ghosting' mechanism in the STK-Mesh library. The custom ghosting STK-mesh interface, which has been deployed to the production code base, is used for a variety of low-Mach use cases including the hybrid CVFEM/DG algorithm, overset, multi-physics coupling over disparate meshes, e.g., fluid structure interaction (FSI), congugate heat transfer (CHT), and the coupling of participating media radiation (PMR) to non-isothermal fluid flow. The custom ghosting interface is also used for data manipulation including the initialization of a new fine mesh simulation using a precursor coarse mesh result, select output to a subset of the mesh (consider a PIV or PLIF-like application), and in conjunction with a line-of-sight data probes post processing interface. STK-Mesh defines a terminology where a 'send-ghost' is an element owned by the current processor and needs to be ghosted onto another processor. The element is called a 'receive-ghost' on that non-owning processor.

Ghosting lists are defined by passing STK-Mesh a list of owned elements and the processors to which they should be sent. Due to sliding mesh motion, the ghosting lists can potentially change every time-step, as new elements move into the opposing space, and also as other elements move on and are no longer opposing. However, much of the time the ghosting lists donot completely change; i.e., not all elements are replaced rather a few are removed as they move out of range. New elements are added as required.

The original implementation discarded all ghosts and recreated ghosting lists every time-step. It was later found that it was more efficient to take an incremental approach where at each time-step we determined 'change-lists' for the ghosting, i.e., a list of new elements to ghost, and a list of ghosted elements that are no longer required. Passing these lists to STK-Mesh allowed for less communication and computational expense as compared to completely destroying and re-creating all ghost information. The methodology outlines a in-situ paradigm to explore in other aspects of the code, e.g., the in-situ construction of the matrix graph.

6.4 Hybrid Mesh Topologies

The mesh created for the V27 contains multiple element topologies, including hexahedra, tetrahedra, pyramids, and wedges. Initial simulation studies with this mesh exhibited some artificially high velocities, depending on the selection of wall boundary conditions and advection stabilization approach. The high velocities were observed at sharp corners in the turbine geometry; for example, Figure 20 shows the spurious behavior occurring at corners of the nacelle. It was noted that the use of wall functions made this behavior worse (relative to using a no-slip boundary condition[5]), and that use of a nonlinear stabilization operator (NSO), in addition to default advection stabilization, improved this behavior.

To explore this issue further, a model problem was constructed, consisting of flow past a rectangular prism, with dimensions approximately equal to those of the V27 nacelle. Various meshes were made for this problem, utilizing different element topologies and different anisotropic mesh spacings near sharp corners. The flow past this rectangular prism was simulated for a free-stream velocity of 7.6 m/s. The nominal simulation parameters were as follows:

- WALE turbulence model

[5] Unit testing of the `Wedge6` noted a mapping error of opposing node to the nearest node to a boundary integration point for a particular element ordinal.

- CVFEM scheme
- Second-order time advancement
- Default advection stabilization
- Shifted gradient operator for pressure
- Wall-function boundary condition on solid surfaces
- Wall-normal mesh spacing for first off-surface mesh point of 0.001 meters

First, a mesh based on near-wall extrusion of surface triangles into wedge elements was examined, in order to reproduce the solution behavior seen in the V27 problem. Figure 21 shows a closeup of this mesh, and the oscillatory streamwise velocity field that develops near corners. To assess the relationship of this solution behavior to element type, a similar mesh was constructed, only with surface quads extruded into hex elements. The surface element edge lengths and mesh spacing distribution in the wall-normal direction were very similar to that of the original wedge mesh. Figure 22 shows that the undesirable solution oscillations occur also for this hex mesh, suggesting that element topology is not to blame.

To examine the effect of spatial resolution of the mesh near corners, two meshes were constructed with refinement of the surface elements in the direction normal to the sharp edges of the geometry. One mesh employed extruded sharp-edge-aligned triangles, with mesh spacing along the surface and normal to the sharp edge equal to 0.001 meters, see Figure 23. The second mesh employed extruded quads with a sharp-edge-normal mesh spacing of 0.002 meters, see Figure 24. Significant spurious oscillations in the streamwise velocity field are not observed for these two meshes, indicating that mesh resolution near the sharp corners plays a fundamental role in the solution behavior.

Figure 25 illustrates the issue that arises when coarse surface meshes are used near a sharp corner with wall function boundary conditions. Consider the flow tangential to the upper surface, which separates from the surface at the sharp corner as shown. Just downstream of the corner, a very thin shear layer develops; the magnitude of the velocity jump across this shear layer is strengthened by the use of a wall function, which allows for a non-zero "slip" velocity at the mesh node coincident with the sharp corner. On the back-side solid surface, the boundary condition enforces zero max flux in the streamwise direction, reinforcing the generation of the thin layer. If the mesh spacing across this thin shear layer is inadequate to properly resolve it, then in the absence of sufficient numerical dissipation it is expected that numerical oscillations will develop.

In order to further explore the effect of numerical scheme on these oscillation, the original extruded wedge mesh with coarse near-corner resolution was also run with the edge-based scheme as well as with the CVFEM scheme with the NSO operator (option NSO_2ND_ALT). Figure 26 shows that the edge scheme does exhibit some spurious oscillations, but the magnitude is smaller than those of the CVFEM scheme. It is not clear why this is the case; however, we conjecture that the edge scheme is more dissipative when faced with under-resolution of the mesh at a corner. The NSO solution is smooth and stable, but also more diffuse, as seen in Figure 26. Since the NSO operator is based on a fine-scale residual estimate, the scheme is able to apply more dissipation to an under-resolved flow feature and maintain numerical stability.

Finally, we note that artificially high velocities were also observed near blade tips. Figure 27 shows the mesh topology at a blade tip, illustrating the same issue encountered with the nacelle: under-resolution of the separation zone near a sharp geometric feature. An effort is underway to improve the meshes in these areas, such that adequate resolution of thin separation layers is obtained.

7. CONCLUSION

In this milestone effort, a design-order hybrid CVFEM/DG scheme was developed, verified and deployed to support the wind application space that requires dynamic sliding mesh interfaces. Both low- and high-order support was demonstrated for the canonical viscous vortex simulation. A set of numerical tests to drive the improvement of the hybrid scheme at hybrid mesh topology nonconformal interfaces was developed. It is hoped that the research carried out under this milestone effort (and captured in a technical paper) will be accepted to the Journal of Computational Physics (at the time of this milestone write-up, the paper is in "review" status).

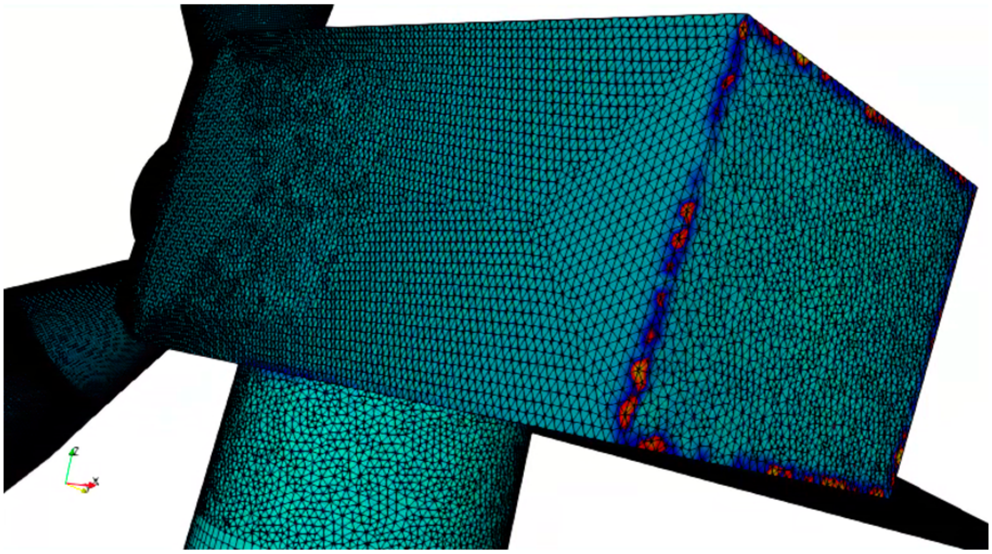
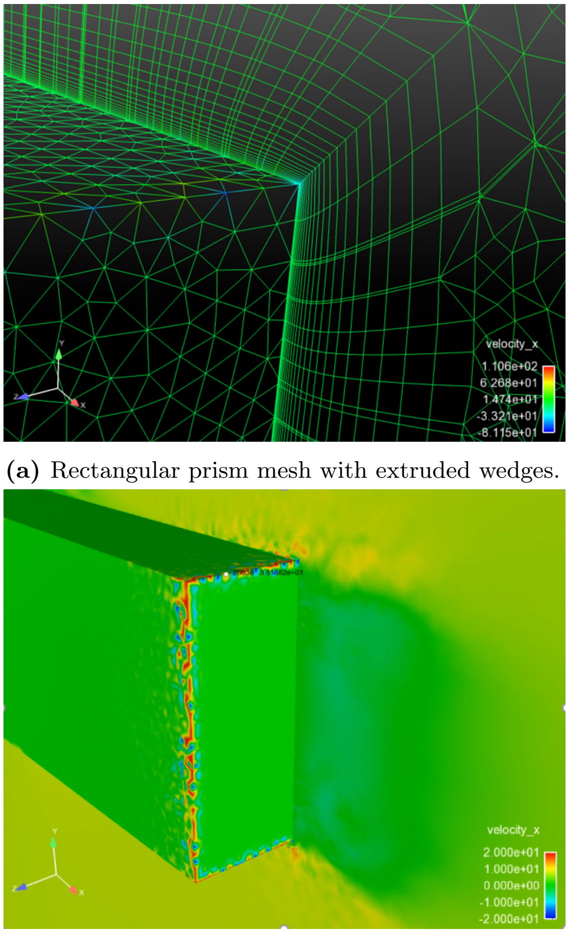


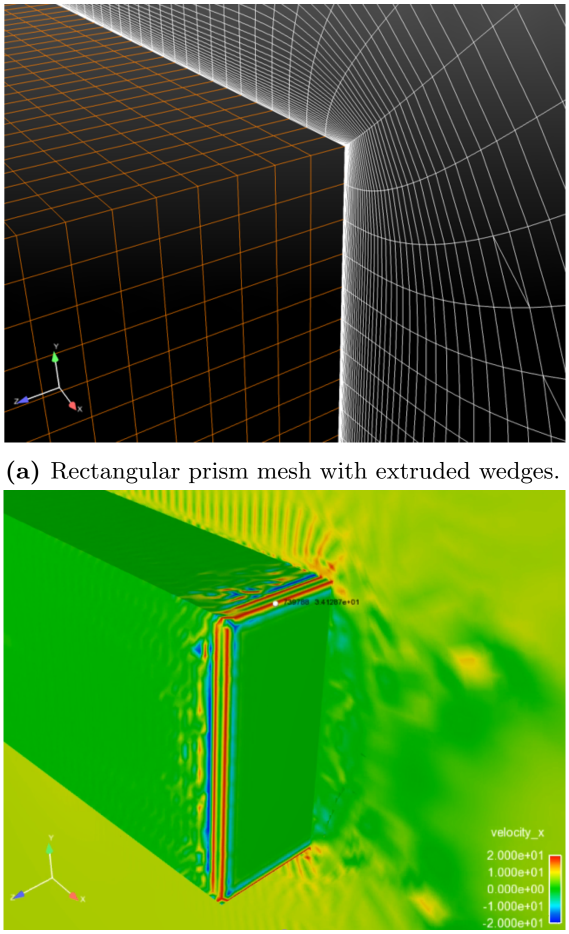
Figure 20: Spurious velocities at sharp corners of the V27 nacelle.



(a) Rectangular prism mesh with extruded wedges.

(b) Stream-wise velocity solution.

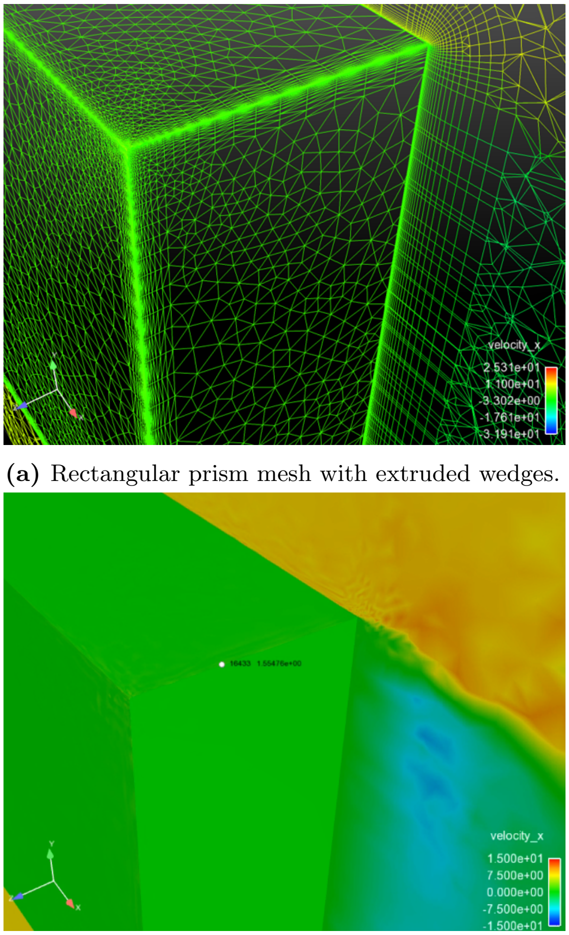
Figure 21: Rectangular prism test problem, CVFEM with wedge elements.



(a) Rectangular prism mesh with extruded wedges.

(b) Stream-wise velocity solution.

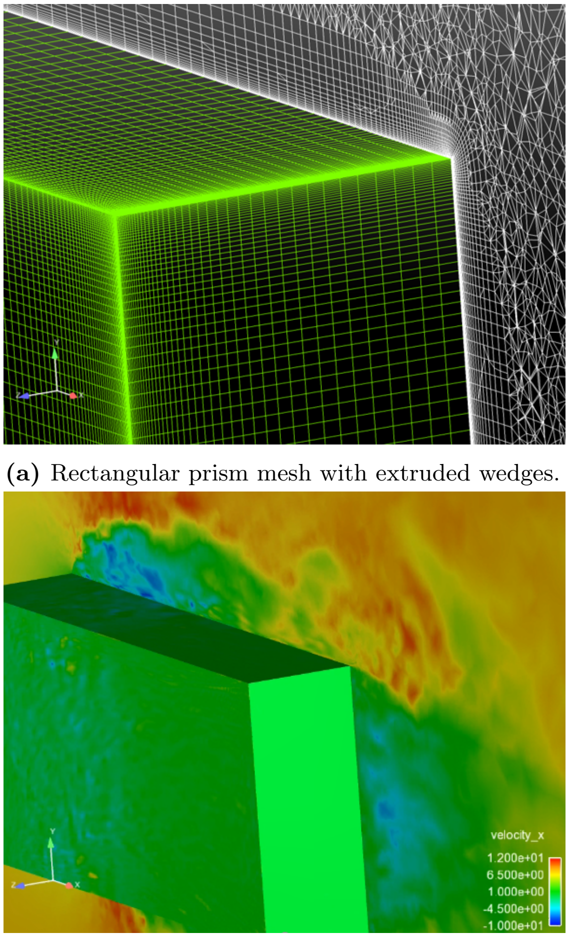
Figure 22: Rectangular prism test problem, CVFEM with hex elements.



(a) Rectangular prism mesh with extruded wedges.

(b) Stream-wise velocity solution.

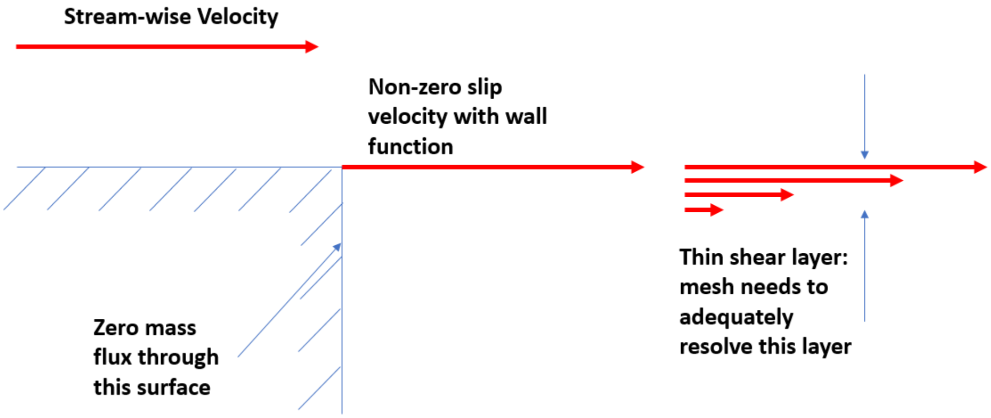
Figure 23: Rectangular prism test problem, CVFEM with wedge elements and refinement near sharp corners.



(a) Rectangular prism mesh with extruded wedges.

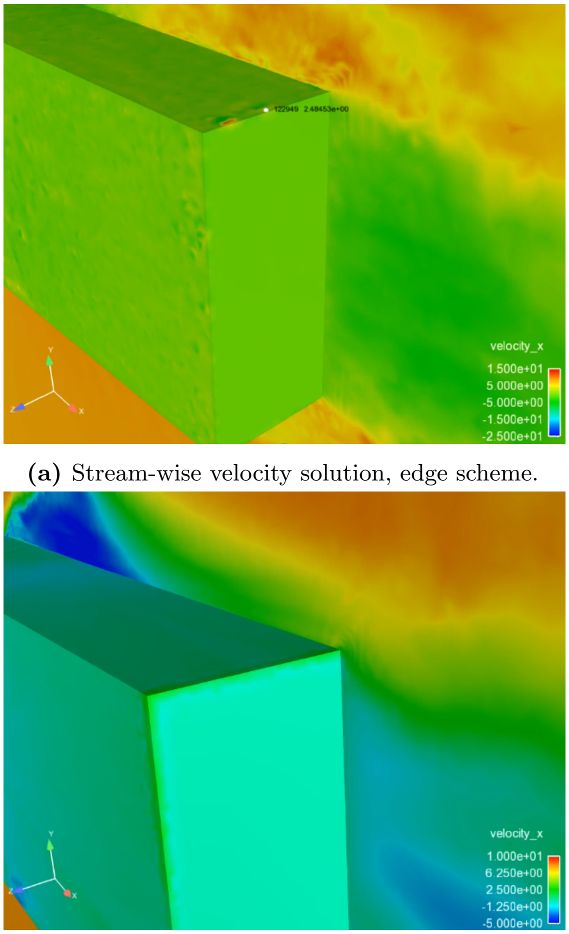
(b) Stream-wise velocity solution.

Figure 24: Rectangular prism test problem, CVFEM with hex elements and refinement near sharp corners.



Stream-wise Velocity
Non-zero slip velocity with wall function
Thin shear layer: mesh needs to adequately resolve this layer
Zero mass flux through this surface

Figure 25: Schematic illustrating the under-resolution issue at sharp corners.



(a) Stream-wise velocity solution, edge scheme.

(b) Stream-wise velocity solution, NSO stabilization term.

Figure 26: Rectangular prism test problem on the coarse wedge mesh, with alternative spatial discretization options.

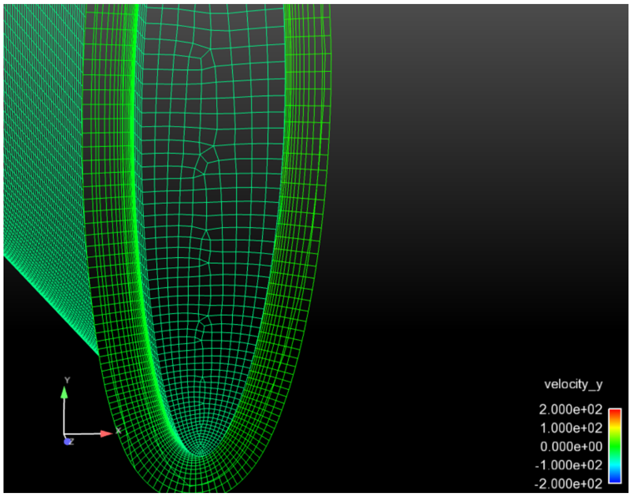


Figure 27: V27 blade tip mesh topology, illustrating coarse resolution near sharp edges.

Several production runs have been carried out using the hybrid sliding mesh scheme. Production runs for the Vestas V27 turbine (with and without tower/nacelle) have been run at modest hybrid mesh resolutions, i.e., 60 to 500 million elements. The project also drove a stretch goal of running the 1.3 billion element V27 production mesh. Application-driven improvements noted the testing of several modified matrix assembly procedures to mitigate matrix initialization requirements. The NGP efforts realized in FY17/Q4 were transitioned to production status and were demonstrated within hybrid mesh V27 production runs.

The FY18/Q1 project effort also drove the quantification of solver and set-up performance. Moreover, a new internal matrix assembly procedure for the Hypre solver package was developed. Results indicate that matrix reinitialization costs are increasing for Tpetra-based assembly while for the Hypre interface, these costs are remaining low. Preconditioner setup costs at the scales tested were reasonable, although previous scaling data suggests that at mesh counts beyond the 32-bit scale, these costs will increase.

7.1 Path Forward

A set of paths to explore in future ExaWind milestone efforts are as follows:

1. Given the INCITE computational award, it is anticipated that the scale of simulation meshes will increase to beyond the 32-bit limit. This will be a priority for the remaining FY18 effort.
2. Continued quantification of re-initialization timing events beyond the 1.3 billion element mesh resolution (at present, the team has also generated a nearly five-billion element hybrid mesh).
3. Improved matrix initialization approaches such as static graph construction will be prioritized (at the time of this report, this is an active story under Jira).
4. Advance matrix stencil bloating for the nonconformal interface in order to reduce the frequency of matrix setup steps while maintaining the full Jacobian entries.
5. Continued exploration of ATDM-based parallel searches.
6. Dynamic, intelligent search tolerances to ensure efficient coarse searchers. As the search is processed each time step, an appropriate length scale from the previous search can be used to guide the current search.
7. Combined story between the ExaWind project and Sandia National Laboratories Thermal/Fluids department (1541) to drive CVFEM quadrature placement.
8. Continued work on the NSO kernel.

8. ACKNOWLEDGEMENTS

The ExaWind project would like to acknowledge Dr. Chris Bruner (Aerosciences department, SNL) for his work in meshing the V27 turbine configuration. The project would also like to thank Dr. Tyler Voskuilen (Computational Thermal/Fluid, SNL) for his idea of matrix bloating and his foundational unit testing of the `Tet4SCS` Master element implementation in the ASC Sierra Fuego low-Mach module.

REFERENCES

[1] D. Arnold, F. Brezzi, B. Cochburn, and D. Marini. Unified analysis of the discontinous Galerkin method for eliptic problems. *SIAM J. Numer. Anal.*, 39:1749–1779, 2002.

[2] H. De Sterck, Yang U. M., and Heys J. J. Reducing complexity in parallel algebraic multigrid preconditioners. *SIAM Journal on Matrix Analysis and Applications*, 27:1019–1039, 2006.

[3] S. Domino. Github: NaluCFD/Nalu. https://github.com/NaluCFD/Nalu. [Online; accessed 21 Sept. 2017].

[4] S. Domino. A comparison between low-order and higher-order low-Mach discretization approaches. In Parviz Moin and Javier Urzay, editors, *Studying Turbulence Using Numerical Simulation Databases - XV*, pages 387–396. Stanford Center for Turbulence Research, 2014.

[5] S. Domino. Sierra low Mach module: Nalu theory manual – version 1.0. Technical Report SAND2015-3107W, Sandia National Laboratories, 2015.

[6] S. Domino. Design-order, non-conformal low-Mach fluid algorithms using a hybrid cvfem/dg approach. *Submitted to J. Comp. Phys.*, 2017.

[7] J. Guermond, A. Larios, and T. Thompson. Validation of an entropy-viscosity model for large eddy simulation.). In J. Frohlich, editor, *Direct and Large-Eddy Simulation IX*, Switzerland, 2015. Springer International Publishing.

[8] J. Guermond, R. Pasquetti, and B. Popov. Entropy viscosity method for nonlinear conservation laws. *J. Comp. Phy.*, 230(11):4248–4267, 2011.

[9] J. Hu, A. Prokopenko, C. M. Siefert, R. S. Tuminaro, and T. A. Wiesner. MueLu multigrid framework. http://trilinos.org/packages/muelu, 2014.

[10] K. W. McAlister and R. K. Takahashi. Naca 0015 wing pressure and trailing vortex measurements. Technical Report NASA TP-3151, NASA, 1991.

[11] G. Schneider and M. Raw. A skewed, positive influence coefficient upwinding procedure for control-volume-based finite-element convection-diffusion computation. *Num. Heat Trans.*, 9:1–26, 1986.

[12] F. Shakib, T. Hughes, and Z. Johan. A new finite element formulation for computational fluid dynamics: X. the compressible Euler and Navier-Stokes equations. *Comp. Meth. Appl. Mech. Engr.*, 89:141–219, 1991.

[13] B. Zhang. *A high-order computational framework for simulating flows around rotating and moving objects.* PhD thesis, George Washington University, 2017.