Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 00 (2016) 000–000

Procedia Engineering

www.elsevier.com/locate/procedia

26th International Meshing Roundtable

First benchmark of the Unstructured Grid Adaptation Working Group

Daniel Ibanez[a,*], Nicolas Barral[b], Joshua Krakos[c], Adrien Loseille[d], Todd Michal[c], Mike Park[e]

[a]*Sandia National Laboratories, P.O. Box 5800, Albuquerque, NM 87185-1321, United States*
[b]*Department of Earth Science and Engineering, Imperial College London, South Kensington Campus, London SW7 2AZ, United Kingdom*
[c]*The Boeing Company, St. Louis, MO, United States*
[d]*INRIA Saclay-île-de-France, Bâtiment Alan Turing, 91120 Palaiseau, France*
[e]*NASA Langley Research Center, Mail Stop 128, Hampton, VA 23681, United States*

Abstract

Unstructured grid adaptation is a technology that holds the potential to improve the automation and accuracy of computational fluid dynamics and other computational disciplines. Difficulty producing the highly anisotropic elements necessary for simulation on complex curved geometries that satisfies a resolution request has limited this technology's widespread adoption. The Unstructured Grid Adaptation Working Group is an open gathering of researchers working on adapting simplicial meshes to conform to a metric field. Current members span a wide range of institutions including academia, industry, and national laboratories. The purpose of this group is to create a common basis for understanding and improving mesh adaptation. We present our first major contribution: a common set of benchmark cases, including input meshes and analytic metric specifications, that are publicly available to be used for evaluating any mesh adaptation code. We also present the results of several existing codes on these benchmark cases, to illustrate their utility in identifying key challenges common to all codes and important differences between available codes. Future directions are defined to expand this benchmark to mature the technology necessary to impact practical simulation workflows.

1. Introduction

Continued advancements in both computers and algorithms have revolutionized the analysis and design processes for aerospace vehicles though Computational Fluid Dynamics (CFD) tools. CFD simulation places unique demands on the grids required for accurate discretization and solution that are not required for other classes of physical modeling. Alauzet and Loseille [1] document the dramatic progress made in the last decade for solution adaptive methods that include the anisotropy to resolve simulations with shocks and boundary layers. Remaining challenges are identified by the application of these solution adaptive techniques. Park et al. [2] document the current state of solution based anisotropic grid adaptation and motivate further development with the impacts improved capability would have on

aerospace analysis and design. This focus on unstructured grid adaptation is provided in the broader context of the CFD Vision 2030 Study by Slotnick et al. [3]. The Vision Study provides a number of case studies to illustrate the current state of CFD capability and capacity and the potential impact of emerging High Performance Computing (HPC) environments forecast in the year 2030. A key finding of the study is that, "Mesh generation and adaptivity continue to be significant bottlenecks in the CFD workflow, and very little government investment has been targeted in these areas." [3] A set of benchmark cases are documented in this article, which present a framework to evaluate currently available anisotropic grid adaptation methods. This allows the strengths and deficiencies in current tools to be shared and sets the stage for targeted research via a community of developers. These benchmarks are readily available to foster collaboration between established international researchers and new entrants into this research topic. This will enhance the exchange of ideas between industry, academia, and government researchers through collaboration and accelerate development to address these bottlenecks.

The encouragement of new entrants is key to making anisotropic grid adaptation technology ubiquitous and impacting practitioner workflows. Appendix C of Park et al. [2] addresses the critical adoption piece of this technology and advocates the need for multiple implementations, because technology diffusion research has identified the number of institutions that make a firm entry of a product into the market, is a stronger driver than the strengths of an individual product for new technology adoption [4].

These benchmark cases are a continuation of the efforts of Park et al. [5] to decompose the solution adaptive process into a number of subprocesses that can be independently verified, evaluated, and improved. Developing and documenting the evaluation methods is equally important as the test cases themselves. This first version of the benchmarks focus on the grid adaptation mechanics. Extensions of the benchmarks are envisioned that examine error estimation, which will continue as an acute need for efficiency and robustness of grid adaptation. An example of this acute need is that Michal et al. [6] show that the lack of a reliable error estimate can reduce the efficiency of advanced automated anisotropic grid adaptation methods. This first benchmark or its extensions could also become a precursor to a workshop (e.g., Levy et al. [7]) as the size of the solution adaptive community grows.

Providing a benchmark of anisotropic grid adaptation tools to the greater community allows comparing different implementations to understand the implications of implementation choices. This verification by comparison approach is also employed by the Turbulence Modeling Resource Website [8]. "What makes the current website unique is that it focuses on providing ready access to equations, grids, and flow solution details from previously verified codes as an aid to users who wish to verify their own implementations of models on relatively simple cases" [8]. The goal of this work is to define a framework for rigorous examination of anisotropic grid adaptation methods that can guide the implementation and further development and adoption of solution adaptive methods.

2. Benchmarks Site

A central repository for the UGAWG has been established on GitHub [9]. This site houses the data necessary to set up and run benchmark test cases along with results from various adaptation tools. The top level of the site is divided into *adapt-benchmarks* and *adapt-results* repositories. The *adapt-benchmarks* section contains a collection of test cases that UGAWG members can use to evaluate meshing tools and includes geometry definitions, initial meshes and metric fields for each case. The *adapt-results* section contains results generated and uploaded by various UGAWG participants using several different meshing tools. These benchmark cases and the associated results provide the opportunity to evaluate, compare and contrast meshing tools with other contemporary tool sets. The Git version control system [10] and the GitHub website have become vehicles for collaboratively contributing to open source software projects. In many ways, the central repository for the UGAWG is leveraging this software ecosystem to lower the barriers to contribution and encourage new entry into unstructured grid adaptation research.

2.1. Geometry Models

The benchmarks currently contain two models. The geometry for the first benchmark case is represented by a unit cube. This case was selected to evaluate a meshing tools ability to match a prescribed metric field in the absence of surface curvature. This case provides the opportunity to evaluate the metric conformance without introducing geometry projection or surface curvature complications. By removing geometry projection issues, this cube test case

can easily be completed by most adaptive remeshing tools, which provides a common baseline for comparison. The second geometry model subtracts from the unit cube a cylinder of radius 0.5 oriented along the z-axis positioned at $x = 0$, $y = 0$. This case tests the ability of the meshing tool to build a mesh that conforms to a metric field while simultaneously maintaining the geometry shape. The geometry definition is provided in STEP and Electronic Geometry Aircraft Design System (EGADS [11]) formats for remeshing tools that have an embedded geometry modeling kernel. Due to the simple shapes of the Cube or Cube-Cylinder, geometry can also be represented analytically.

2.2. Metric Distribution

In addition to the geometry definition, each benchmark test case includes a prescribed mesh sizing distribution. The sizing distribution is provided in the form of analytically defined metric functions. Three metric distributions have been defined that represent anisotropic features commonly found in computational analysis problems. The first is a linear function with anisotropic stretching centered about the $z = 0.5$ plane.

The linear metric field is described by:

$$M = \begin{bmatrix} h_x^2 & 0 & 0 \\ 0 & h_y^{-2} & 0 \\ 0 & 0 & h_z^{-2} \end{bmatrix}, \tag{1}$$

where $h_x = 0.1,\ h_y = 0.1,\ h_0 = 0.001$ and $h_z = h_0 + 2(0.1 - h_0)|z - 0.5|$. This metric field is representative of a shear layer in the absence of curvature and will be referred to as the Linear-1.

The second metric field is described by:

$$M = \begin{bmatrix} \cos(t) & -\sin(t) & 0 \\ \sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_r^{-2} & 0 & 0 \\ 0 & h_t^{-2} & 0 \\ 0 & 0 & h_z^{-2} \end{bmatrix} \begin{bmatrix} \cos(t) & \sin(t) & 0 \\ -\sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2}$$

where $r = \sqrt{x^2 + y^2},\ t = \text{atan2}(y, x),\ h_z = h_t = 0.1,\ h_0 = 0.001$ and $h_r = h_0 + 2(0.1 - h_0)\,|r - 0.5|$. Where t is in the θ direction and r is the radial direction. This metric field represents a curved shear layer positioned with the curved surface of the Cube-Cylinder geometry. This metric distribution will be referred to as Polar-1.

A slight modification to the Polar-1 distribution is also used and is defined as:

$$d = 10\,(0.6 - r), \quad \text{and} \quad h_t = \begin{cases} 0.1 & \text{if } d < 0 \\ d/40 + 0.1(1 - d) & \text{if } d \le 0 \end{cases} \tag{3}$$

This modified Polar distribution has lower gradation and is easier to satisfy with high-quality elements by refining in the θ-direction near the layer. It is the Polar-2 distribution.

2.3. Test Cases

Three benchmark test cases have been defined for this paper. The first case consists of the Cube geometry with the Linear-1 metric distribution. This baseline benchmark tests the ability of a meshing tool to build a mesh that conforms to a metric distribution in the absence of geometry or metric curvature. Metric conformance is measured by comparing the edge lengths of the resulting mesh with the prescribed metric distribution. Poor levels of metric conformance could indicate a fundamental problem with the implementation of the meshing operators and should be investigated before moving onto the other benchmark cases. The remaining benchmark cases combine the Cube-Cylinder geometry model and the Linear-1, Polar-1, and Polar-2 distributions. The Cube-Cylinder with Linear-1 introduces geometry curvature away from the presence of a highly anisotropic metric field and is a good measure of a meshing tools geometry preservation capability. The next benchmark cases consisting of the Cube-Cylinder model and Polar-1 begins to explore the interaction between curved geometry and a highly anisotropic geometry aligned metric field. This case is representative of attached shear layers commonly found in fluid flow problems. The final benchmark consisting of the Cube-Cylinder and Polar-2 distribution is similar to the objectives of the previous benchmark but is easier to achieve higher quality elements in the adapted mesh due to less aggressive gradation.

Evaluation criteria for all test cases is described in detail in Section 4 and includes metric conformance as measured by edge length relative to the prescribed metric (see Equation 4), as well as an element quality measure in metric space (see Equation 6). Each participating code took slightly different approaches, and Section 3 will describe these approaches in as much detail as is possible within article length constraints.

2.4. Evaluation Framework

To provide a consistent evaluation of meshes, a common set of tools was used to measure the metric conformance and other evaluation criteria on all meshes. The libMeshb [12] format was chosen as the common file format for transferring mesh data between the meshers and the mesh evaluation tool. Each mesh was read into the evaluation tools and compared against the analytic metric field. The tools produce three kinds of output: a rendering of the mesh using ParaView [13], histograms of metric edge lengths and metric element qualities, and maximum/minimum length and quality values. A comparison and analysis of the resulting evaluation data is provided in Section 4.

3. Participating Codes

UGAWG members provided grid adaptation codes that are the result of industry, academic, and governent investment and development. Some are open source, which allows for detailed examination of implementation details.

3.1. EPIC

EPIC is a Boeing internally developed grid adaptation tool that combines local edge break, edge collapse, element reconnection and node smoothing operators [6, 14]. EPIC development has focused on industrial aerospace CFD applications with emphasis on robust handling of complex geometry and efficient use of parallel computing resources. Edge lengths in EPIC are computed with numerical integration of the metric field along an edge instead of assuming a linear variation of the metric M in log-Euclidean space as shown in Equation 4. EPIC grids using three sets of mesh operators are presented: only insertion and collapse (EPIC-IC); insertion, collapse, and swap (EPIC-ICS); insertion, collapse, swap, and node movement (EPIC-ICSM).

3.2. refine

Adaptive grid results from two versions of the refine tool are presented. The refine software package is developed and distributed by NASA. Both versions are designed to produce a unit mesh [15] in a provided metric field. The original version, refine/one, is documented by Park and Darmofal [16]. The current version under development, refine/two, uses the combination of edge split and collapse operations proposed by Michal and Krakos [14]. Node relocation is performed to improve adjacent element quality. A new ideal node location of the node is created for each adjacent element. A convex combination of these ideal node locations is chosen to yield a new node location update that improves the element shape measure in the anisotropic metric [17]. Geometry is accessed through EGADS application program interface.

3.3. Omega_h

Omega_h is an open-source grid adaptation library [18–20], developed by Rensselaer Polytechnic Institute and subsequently by Sandia National Laboratories. Like the other codes in this study, it aims to be a state-of-the-art implementation of grid adaptation by local topological modifications. Omega_h has certain unique objectives: First, it targets tightly coupled adaptivity within a simulation, which requires remapping the solution accurately. This motivates minimizing the number of modifications. Second, it targets simulations outside the CFD space, including solid mechanics and shock hydrodynamics. This motivates a much stronger focus on element quality and efficient operation with isotropic metrics. Third, it targets high performance execution using threading and even GPUs.

The core algorithm in Omega_h consists of one loop of alternating edge splitting and edge collapsing to satisfy length, followed by another loop that uses edge swapping and edge collapsing to satisfy quality. Snapping to geometry

(using EGADS) is part of the second (quality) loop: a step is added, which moves all nodes as far as they can toward the snapping goal, followed by swapping/collapsing to correct shapes. To accommodate highly anisotropic target metrics, Omega_h applies its full adaptive algorithm several times, where the metric used each time is an interpolated metric between the original implied metric and the final desired metric. In both snapping and metric approaching, the criteria that determines the step size is element quality (the step is halved until all elements are above a minimum quality in the interpolated metric space). For all results presented, the minimum quality threshold was set to 30% (see Equation 6).

3.4. Pragmatic

Pragmatic [21, 22] is an open source 2D and 3D anisotropic remesher developed as a C++ library at Imperial College London. Initially targeted at geophysical flow simulations, it now aims at generating quality meshes for a wide range of numerical simulations. It has been integrated into the PETSc library [23, 24].

The adapted mesh is obtained from the input mesh through a series of local mesh manipulations. Iterative applications of coarsening (edge collapse), edge/face swapping and refinement (edge splitting) first optimize the resolution and the quality of the mesh, followed by a final quality-constrained Laplacian smoothing step that fine-tunes the mesh quality. The element internal quality function that is optimized is the functional defined in Vasilevskii and Lipnikov [25]. Pragmatic was started as a hybrid threads and MPI parallel code. Since then, the enthusiasm for hybrid parallelism has waned on the solver side, so a more classic purely distributed memory approach was favored in Pragmatic.

Whereas adaptation of surfaces based on CAD representation using EGADS is in progress, an ad hoc procedure is used in this paper that projects new vertices onto an analytic surface as soon as they are created. In the Polar-1 Cube-Cylinder case (see Equation 2), a coarse metric in the radial direction ($h_0 = 0.1$) is initially prescribed and then h_0 is progressively reduced to the desired metric in a series of steps.

3.5. feflo.a

Feflo.a is an adaptation code developed at INRIA. It is based on a two-step procedure to generate a unit-mesh [26, 27]. The first step aims at improving the edges length distribution with respect to the input metric field. In its original version, only classical edge-based operators (insertion and collapse) are used during this step. The second step is optimization of the mesh element shape measures with node smoothing and tetrahedra edge and face swaps. Feflo.a can handle nonmanifold surface and/or volume meshes composed of simplicial elements. For the surface mesh adaptation, a dedicated surface metric is used to control the deviation of the metric and surface curvature. This surface metric is then combined with the input metric. New points created on the surface are projected to a (fine) background surface grid and optionally CAD via the EGADS API.

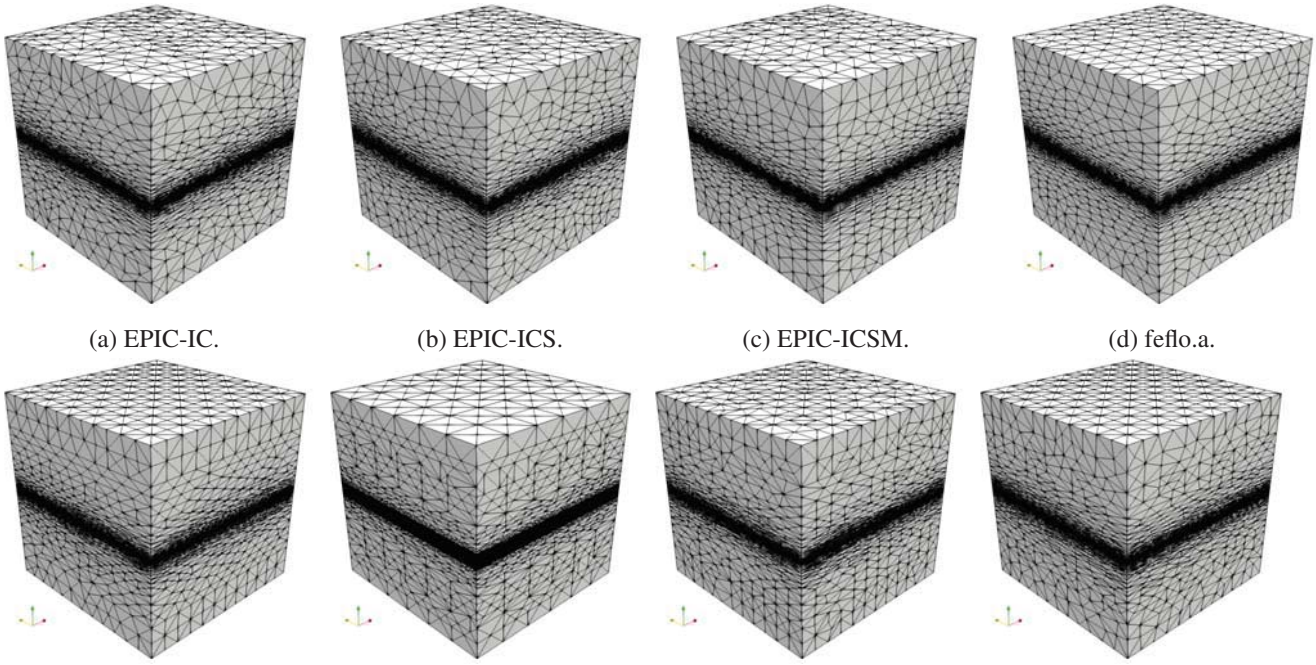
More recently, classical edge-based operators have been replaced by a unique cavity-based operator [28, 29]. This cavity-based operator simplifies code maintenance, increases the success rate of mesh modifications, has a constant execution time for many different local operations, and robustly inserts boundary layer grids [30]. When the cavity operator is combined with advancing-point techniques, it produces metric-aligned and metric-orthogonal meshes [31].

3.6. adaptive process

To mimic the requirements of a solution adaptation process, the analytic metric is evaluated on an input mesh. The adaptation mechanics interpolate the metric from the input mesh during adaptation. The adapted mesh becomes the input mesh and the adaptation process is repeated until the adapted grid and the input mesh are equivalent to edge length statistics. Multiple cycles of metric evaluation are necessary because the initial grid does adequately represent the analytic metric.

4. Results

For this publication, we use two local criteria of metric satisfaction. First, we measure the edge-length criterion as presented by Park et al. [5]. Our formula for edge length is based on an assumption that the logarithm of desired



(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) feflo.a.

(e) Omega_h. (f) refine/one. (g) refine/two. (h) Pragmatic.

Fig. 1. Meshes for the linear cube metric.

length varies linearly along the edge [32] and is defined by

$$L_e = \begin{cases} \frac{L_a - L_b}{\log(L_a/L_b)} & |L_a - L_b| > 0.001 \\ \frac{L_a + L_b}{2} & \text{else} \end{cases}, \tag{4}$$

$$L_a = (v_e^T M_a v_e)^{\frac{1}{2}}, L_b = (v_e^T M_b v_e)^{\frac{1}{2}}. \tag{5}$$

An ideal edge has a metric length of one in a unit mesh. Second, we also examine element quality, using the mean ratio formula for tetrahedra (Equation 6) where K is a tetrahedron, $|K|$ is its volume, v_e is the vector along one of its edges e, and $|\hat{K}|$ is the volume of a tetrahedron with unit edge lengths. $M_{\max}$ is a single metric tensor being used to measure the whole tetrahedron. In this case, we choose $M_{\max}$ as the adjacent vertex metric with largest determinant,

$$Q_K = \frac{\left(\frac{|K| \det(M_{\max})^{\frac{1}{2}}}{|\hat{K}|}\right)^{\frac{2}{3}}}{\frac{1}{6}\sum_{e \in K} v_e^T M_{\max} v_e}, \tag{6}$$

$$M_{\max} = \underset{M_v, v \in K}{\arg\max} \det M_v. \tag{7}$$

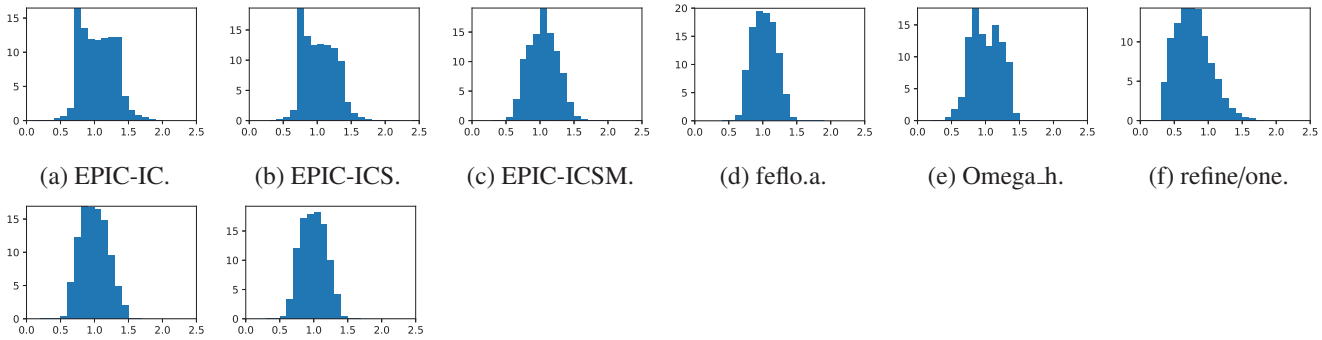
An ideal element has an ideal mean ratio of one. We also consider the global criterion of number of elements, as the main purpose of solution-based adaptivity is to minimize the number of degrees of freedom while maximizing accuracy.

4.1. Linear Cube Case

Table 1 presents the statistics for the satisfaction criteria that each code produced with the linear metric input over the cube domain. In the EPIC family, we see a clear improvement in minimum quality and edge length range as operators are added going from EPIC-IC to EPIC-ICS and EPIC-ICSM. The high maximum length measures for EPIC are likely due to the fact that it does not use Equations 4 and 5 to measure length internally [5]. Omega_h, feflo.a, and refine/one all achieve the same maximum length of 1.80, although feflo.a and Omega_h have higher

Table 1. Criteria statistics for linear cube metric.

Code	Min. Quality	Min. Length	Max. Length	#Elements
EPIC-IC	0.10	0.15	3.48	50860
EPIC-ICS	0.25	0.32	3.05	49262
EPIC-ICSM	0.36	0.39	2.44	45892
feflo.a	0.49	0.45	1.80	45158
Omega_h	0.30	0.21	1.80	51666
refine/one	0.06	0.03	1.80	112543
refine/two	0.05	0.29	1.67	51587
Pragmatic	0.46	0.34	1.67	49332



(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) feflo.a. (e) Omega_h. (f) refine/one.

(g) refine/two. (h) Pragmatic.

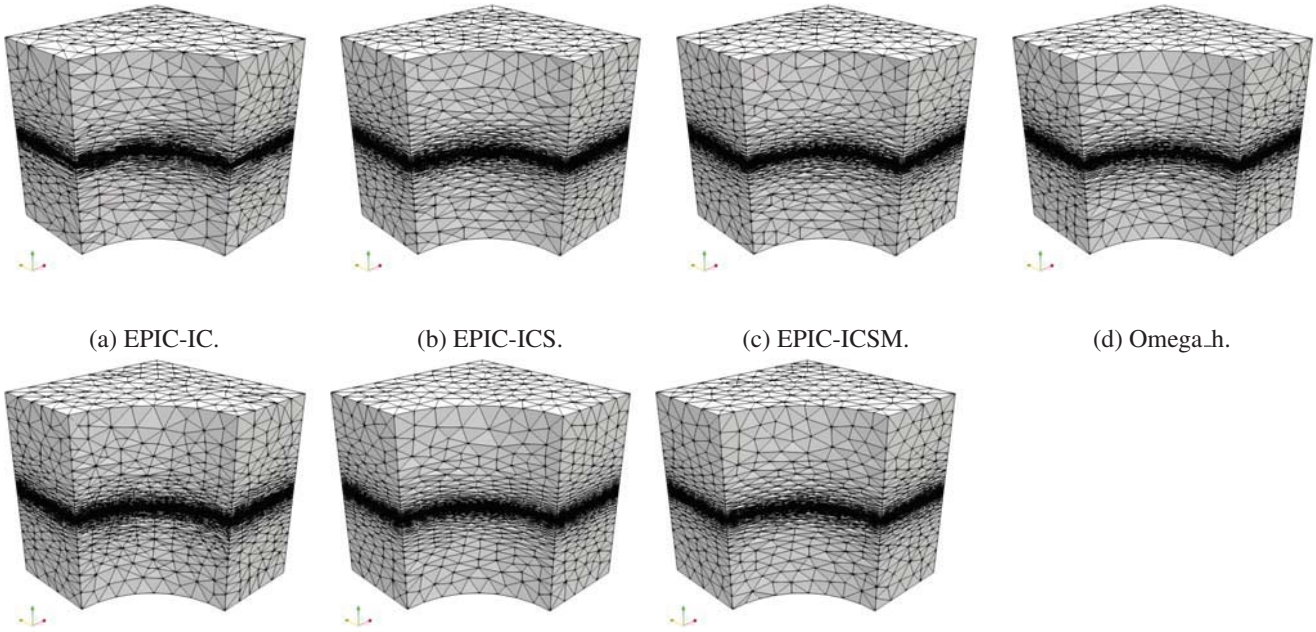
Fig. 2. Length histograms for the linear cube metric.

qualities and minimum lengths. EPIC-IC, EPIC-ICS, and Omega_h all get roughly the same element count (50K), while EPIC-ICSM and feflo.a get slightly better element counts (45K). This is likely due to their use of smoothing (mesh motion), which allows more fine tuning than topology modifications alone. The feflo.a statistics are the best in terms of minimum quality, minimum length, and element count.

Fig. 2 shows a more in-depth look at the edge length distributions for the linear cube metric via histograms. Once again we see a distinction between codes with and without smoothing, with flatter length profiles for EPIC-IC, EPIC-ICS, and Omega_h showing two distinct local maxima. This is likely due to the use of upper and lower thresholds to choose when to refine and coarsen edges. A refinement can be viewed as removing an edge from a high histogram bin and adding several edges to lower bins, accumulating edges in bins just below the threshold. Since the thresholds are typically spaced a factor of two apart, we see the two accumulations of edges. EPIC-ICSM and feflo.a show a much smoother, bell-curve-like distribution with a single maximum. Recall from Table 1 that refine/one produces twice as many elements as the other codes, and the explanation for this can be found in Fig. 2f: it produces many short edges that other codes coarsen. The quality histograms for this case are omitted; all codes showed similar distributions.

4.2. Linear Cube-Cylinder Case

There are three different metric cases with the Cube-Cylinder geometry, which adds geometric curvature as a new difficulty. Table 2 shows minima and maxima for the relevant criteria for each code. The refine/one code is absent for cube-cylinder geometry cases, because it has not implemented the EGADS API for curved geometry resolution. On this geometry, we start to see EPIC-IC perform much worse than other codes, with the longest edge being over 50× longer than desired, the shortest edge being 100× shorter than desired, and the minimum quality being below the output precision of our measurement tools. Fortunately, EPIC-ICS and EPIC-ICSM perform much better, with edge length ranges similar to their cube results. Omega_h maintains its minimum quality at 30%, and has a decent edge length range, comparable to the EPIC codes. While Pragmatic achieves a good maximum length bound, its minimum length and quality are much smaller than they were on the linear cube problem.



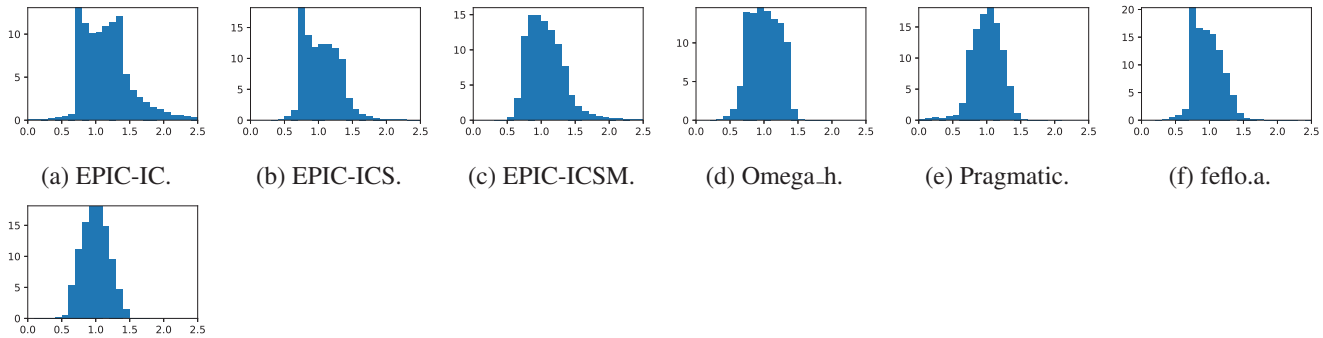
(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) Omega_h.

(e) Pragmatic. (f) feflo.a. (g) refine/two.

Fig. 3. Meshes for the linear cube-cylinder metric.

Table 2. Criteria statistics for linear cube-cylinder metric.

Code	Min. Quality	Min. Length	Max. Length	#Elements
EPIC-IC	< 0.001	0.01	57.65	32711
EPIC-ICS	0.16	0.34	2.83	37481
EPIC-ICSM	0.19	0.32	3.26	34236
Omega_h	0.30	0.29	1.97	40956
Pragmatic	0.01	0.02	2.06	38668
feflo.a	0.04	0.21	2.55	46291
refine/two	0.04	0.16	1.73	38668

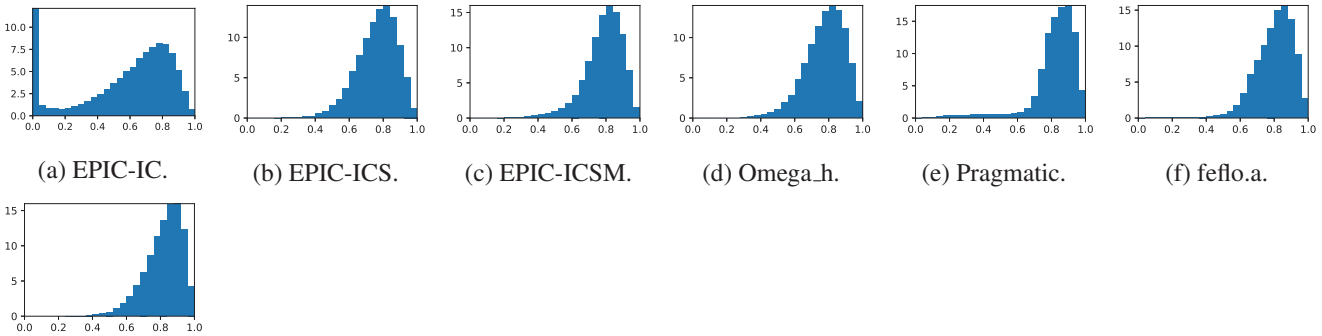


(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) Omega_h. (e) Pragmatic. (f) feflo.a.

(g) refine/two.

Fig. 4. Length histograms for the linear cube-cylinder metric.

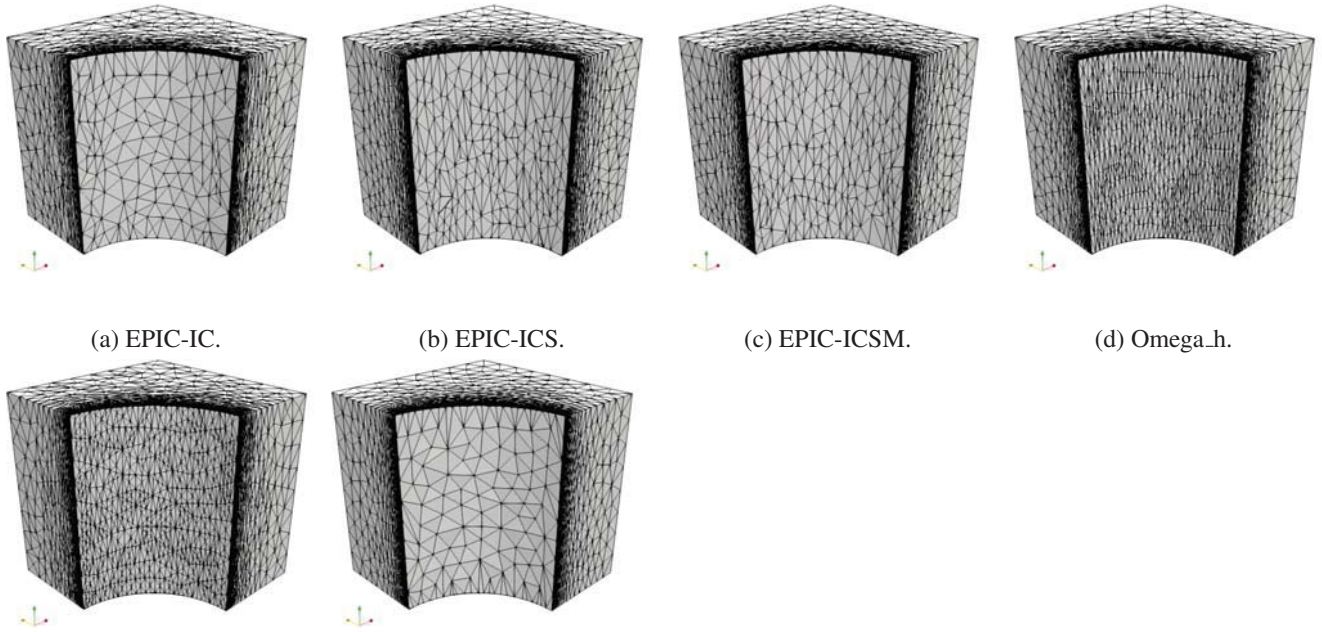
Fig. 4 presents edge lengths as histograms. Both EPIC-IC and EPIC-ICS show two local maxima. At the extrema, EPIC-IC and Pragmatic both have noticeable percentages of their edges in the very low range of $[0, 0.25]$ while all the EPIC codes show a significant tail of edges in the high range $[2.0, 2.5]$. The quality histograms for the linear cube-cylinder metric are shown in Fig. 5. Here we see an interesting property of EPIC-IC on this geometry: a spike of



(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) Omega_h. (e) Pragmatic. (f) feflo.a.

(g) refine/two.

Fig. 5. Quality histograms for the linear cube-cylinder metric.



(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) Omega_h.

(e) Pragmatic. (f) feflo.a.

Fig. 6. Meshes for the Polar-1 Cube-Cylinder metric.

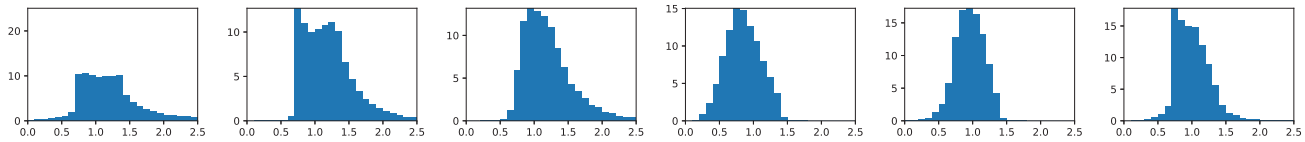
elements in the very low quality range $[0, 0.1]$ which could not be corrected by its limited set of operators. EPIC-ICS and EPIC-ICSM correct this spike, suggesting that swapping is the key shape-correction operator. The histograms of EPIC-ICS, EPIC-ICSM, and Omega_h all look fairly similar, which taper until no significant percentages can be seen below 20%, while Pragmatic has a tail of low-quality elements that continue down to the lowest levels.

4.3. Polar-1 Cube-Cylinder Case

Due to the curvature of the metric specification itself, the main issue in satisfying this metric is high metric gradation. As presented in Table 3, all codes produced a very low minimum quality. Omega_h does not even converge if it cannot find a solution with all elements above 30% quality, so for this metric Omega_h preprocessed the metric using gradation control [32, 33]. The resulting mesh is shown as the result in Fig. 6d. This is what inspired the creation of the Polar-2 metric (see Section 4.4), which is an analytic equivalent of what gradation control did to the metric. In particular, it refines along the tangent direction in order to reduce the rate of metric gradation due to curvature. We still judge the resulting Omega_h mesh by the original Polar-1 metric, hence it technically gets a minimum quality

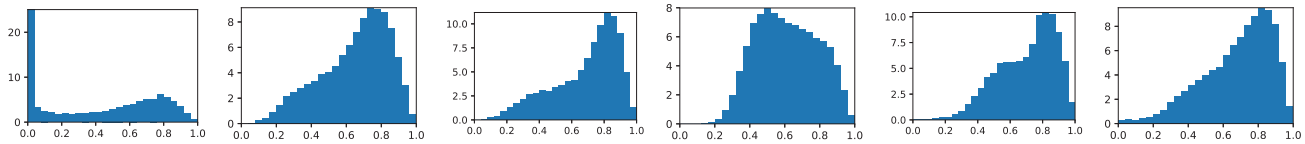
Table 3. Criteria statistics for Polar-1 Cube-Cylinder metric.

Code	Min. Quality	Min. Length	Max. Length	#Elements
EPIC-IC	< 0.001	0.003	67.64	17338
EPIC-ICS	0.05	0.13	6.32	21916
EPIC-ICSM	0.05	0.20	7.45	20237
Omega_h	0.14	0.11	1.71	52235
Pragmatic	0.01	0.02	1.74	33629
feflo.a	0.01	0.18	17.40	35310



(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) Omega_h. (e) Pragmatic. (f) feflo.a.

Fig. 7. Length histograms for the Polar-1 Cube-Cylinder metric.



(a) EPIC-IC. (b) EPIC-ICS. (c) EPIC-ICSM. (d) Omega_h. (e) Pragmatic. (f) feflo.a.

Fig. 8. Quality histograms for the Polar-1 Cube-Cylinder metric.

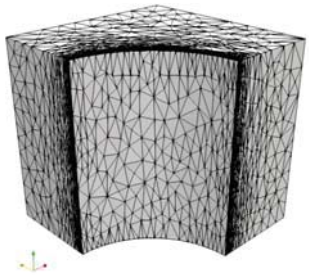
result of 14% here. EPIC-IC continues to show poor performance, while EPIC-ICS and EPIC-ICSM show better performance. EPIC-ICS and EPIC-ICSM have maximum lengths that are approximately twice what they were in the linear cube-cylinder case in Table 2. Pragmatic shows a good maximum length, but has very small minimum lengths and qualities. The curved surface mesh of feflo.a is clearly too coarse (see Fig. 6f), likely due to not selecting the best option for handling this highly graded metric.

Length histograms for the Polar-1 metric in Fig. 7 show similar patterns for the EPIC codes, with the main difference being the reduction of small edges as operators are added. Omega_h and Pragmatic show similar and better controlled distributions, consistent with their maximum edge length results. feflo.a still shows a good distribution, illustrating how histograms alone don't capture important details, and justifying our inclusion of tables of extrema and renderings. In the quality histograms (Fig. 8), EPIC-ICS, EPIC-ICSM, and Pragmatic have similar profiles where frequency increases almost linearly with quality, but noticeable amounts in very low range $[0, 0.1]$. Omega_h has a different distribution, likely with a lower average quality, but a much steeper and more controlled distribution in the low range.

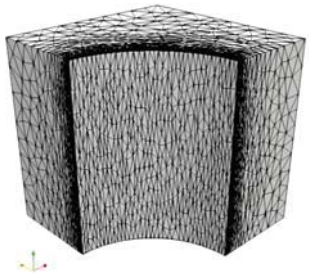
4.4. Polar-2 Cube-Cylinder Case

The modification defining the Polar-2 metric specifically targets a reduced gradation rate, and Table 4 shows that the minimum qualities improved significantly for all codes compared to Table 3 for Polar-1, suggesting a connection between metric gradation rate and the best attainable element quality. Note also that EPIC-ICS and EPIC-ICSM have attained maximum lengths on par with their usual best in the linear cases, suggesting that gradation rate also has an effect on satisfiability of the metric as measured by the length criteria. Omega_h is able to attain its threshold quality of 30%, the best in the group. Finally, note that if one compares the renderings in Fig. 6 to those in Fig. 9, then EPIC-ICS, EPIC-ICSM, and feflo.a are now much more in agreement with Omega_h and Pragmatic.

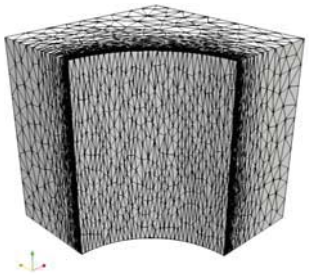
The length histograms for the Polar-2 metric were largely the same as for the Polar-1 metric (Fig. 7), and we omit them for brevity. The quality histograms, on the other hand, show very significant differences between the Polar-1 metric in Fig. 8 and the Polar-2 metric in Fig. 10. All codes (except EPIC-IC) now show a good distribution of quality



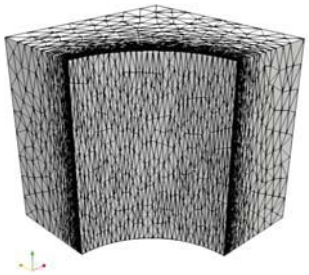
(a) EPIC-IC.



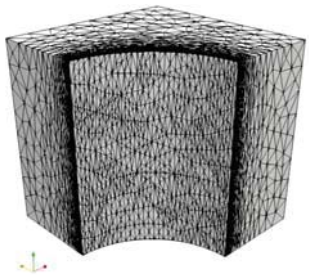
(b) EPIC-ICS.



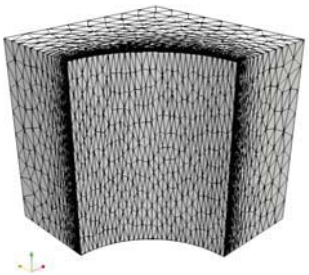
(c) EPIC-ICSM.



(d) Omega_h.



(e) Pragmatic.

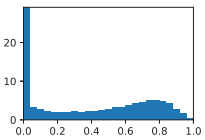


(f) feflo.a.

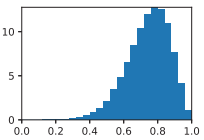
Fig. 9. Meshes for the Polar-2 cube-cylinder metric.

Table 4. Criteria statistics for Polar-2 Cube-Cylinder metric.

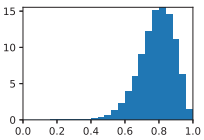
Code	Min. Quality	Min. Length	Max. Length	#Elements
EPIC-IC	< 0.001	< 0.001	70.53	21664
EPIC-ICS	0.15	0.30	3.01	36538
EPIC-ICSM	0.19	0.45	3.04	33417
Omega_h	0.30	0.24	1.81	49151
Pragmatic	0.05	0.12	1.73	47203
feflo.a	0.06	0.18	2.65	53117



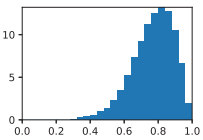
(a) EPIC-IC.



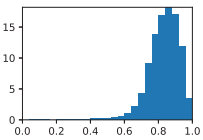
(b) EPIC-ICS.



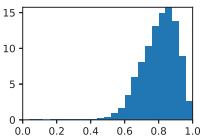
(c) EPIC-ICSM.



(d) Omega_h.



(e) Pragmatic.



(f) feflo.a.

Fig. 10. Quality histograms for the Polar-2 cube-cylinder metric.

centered to the far right with no noticeable percentages in the low range $[0, 0.2]$. Omega_h also shows this nominal distribution, as opposed the strange distribution it had in Fig. 8d. This supports the idea that metric gradation has a significant effect not just on minimum attainable quality but on the qualities of elements all throughout the mesh.

5. Future Directions

Unstructured mesh adaptation has proven to be a reliable tool to predict complex phenomena with complex geometries [1, 6] for steady and unsteady flow regimes. The adaptive process is an iterative procedure where the mesh and the solution are updated to reach *an optimal* mesh solution coupled with a desired level of accuracy. We decompose this process into: (1) error estimation, (2) mesh generation, and (3) solution computation. However, the full benefit of

adaptivity is then achieved only when (1), (2), and (3) are optimally combined. This first set of benchmarks focus on analytic metric-conformity for a simple geometry. We list the main issues that will be addressed by the next sets of test cases for the verification and validation of mesh generation:

Surface mesh adaptation. Surface mesh adaptation becomes critical when an initial mesh is used as in the local remeshing approach. This is even more critical when a boundary layer or highly anisotropic areas are present in the initial mesh and need to be modified. Indeed, due to the presence of the volume mesh near the surface mesh, refining an element may lead to the creation of a negative volume element. This high level of anisotropy has to be then correctly blended with the surface approximation estimate in order not to create numerical artifacts on the geometry. The quality, level of anisotropy, robustness and CPU time need to be assessed for the algorithms described in this paper. We intend to provide test cases starting from more complex initial meshes and with a very high level of anisotropy near the surface. These challenging initial meshes and metrics assess the robustness of the surface mesh adaptation component with respect to the initial starting mesh and the level of anisotropy.

CAD integration. For industrial applications, the use of CAD data is crucial as many quantities of interest depend on the geometry. Providing high quality surface mesh becomes a mandatory feature. We intend to define simple test cases featuring one typical CAD issue at a time: missing topology, CAD tolerance to edges larger than the required mesh size, and highly skewed parameterization.

Adaptive boundary layer. The generation of a boundary layer mesh is generally designed for isotropic surface grids, and is generated only once [30, 34]. Consequently, the boundary layer is frozen while adapting the outer part of the domain [35]. However, this frozen boundary layer mesh strategy is insufficient when the boundary layer interacts with other anisotropic features. The design and assessment of algorithms that are well suited to quickly generate boundary layer meshes in the presence of anisotropic surface meshes is necessary. Test cases will be designed with: (i) an analytical boundary layer metric, and (ii) a solution-based boundary layer metric.

Parallel environment. Finally, the (potential) integration into an HPC environment [36] of each previous mesh refinement techniques needs to be studied. We intend to revisit all the database test cases in a parallel environment. In particular, we can discuss the quality of the generated parallel grids with respect to the sequential one, and also assess the performance advantage of the parallel mesh generation (for surface and volume).

In a more general setting, we then intend to extend progressively the number of test cases and results to more complex geometries and metrics following the discussion of Park et al. [2]. We will apply the same approach as in the preliminary study [5], where only one component is modified (for instance, the error estimate) while the remaining components are kept unchanged (e.g., flow solver, mesh generation algorithm).

6. Conclusion

These benchmark cases have revealed a surprising number of useful insights both into the qualities of participating codes and the nature of mesh adaptation in general. The polar metrics illustrate how metric gradation can make the metric difficult to satisfy, and how gradation control improves metric conformance across all the participating codes. We see that the use of a different edge length criteria by EPIC tends to produce longer edges and fewer elements compared to the other codes, which is important to know when specifying metrics to a certain code. The linear cube metric shows us that in certain cases using nodal repositioning as EPIC and feflo.a do can increase element quality beyond what is feasible with topology modification alone, while the cube-cylinder cases show that preventing low-quality modifications, as Omega_h does, better controls the worst element quality in the more difficult cases.

The Unstructured Grid Adaptation Working Group hopes these and future benchmarks can serve as a common reference point for research in our field. We invite others to apply the benchmark to their codes and submit their results to the repository. Ideas for improvements or additions to the benchmark cases are also welcome.

References

[1] F. Alauzet, A. Loseille, A decade of progress on anisotropic mesh adaptation for computational fluid dynamics, Computer-Aided Design (2015) 13–39.

[2] M. A. Park, J. A. Krakos, T. Michal, A. Loseille, J. J. Alonso, Unstructured Grid Adaptation: Status, Potential Impacts, and Recommended Investments Toward CFD Vision 2030, AIAA Paper 2016–3323, 2016.

[3] J. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D. Mavriplis, CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences, NASA CR-2014-218178, Langley Research Center, 2014. doi:2060/20140003093.

[4] R. Agarwal, B. L. Bayus, The market evolution and sales takeoff of product innovations, Management Science 48 (2002) 1004–1041.

[5] M. A. Park, A. Loseille, J. A. Krakos, T. Michal, Comparing Anisotropic Output-Based Grid Adaptation Methods by Decomposition, AIAA Paper 2015–2292, 2015.

[6] T. R. Michal, D. S. Kamenetskiy, J. Krakos, M. Mani, R. S. Glasby, T. Erwin, D. Stefanski, Comparison of Fixed and Adaptive Unstructured Grid Results for Drag Prediction Workshop 6, AIAA Paper 2017–961, 2017.

[7] D. W. Levy, K. R. Laflin, E. N. Tinoco, J. C. Vassberg, M. Mani, B. Rider, C. L. Rumsey, R. A. Wahls, J. H. Morrison, O. P. Brodersen, S. Crippa, D. J. Mavriplis, M. Murayama, Summary of data from the fifth computational fluid dynamics drag prediction workshop, AIAA Journal of Aircraft 51 (2014) 1194–1213.

[8] C. L. Rumsey, B. R. Smith, G. P. Huang, Description of a Website Resource for Turbulence Modeling Verification and Validation, AIAA Paper 2010–4742, 2010.

[9] M. Park, D. Ibanez, N. Barral, J. Krakos, A. Loseille, T. Michal, UGAWG GitHub site, 2017. `https://github.com/UGAWG`.

[10] S. Chacon, B. Straub, Pro Git, 2nd ed., Apress, Berkely, CA, USA, 2014.

[11] R. Haimes, M. Drela, On The Construction of Aircraft Conceptual Geometry for High-Fidelity Analysis and Design, AIAA Paper 2012–683, 2013.

[12] L. Maréchal, libMeshb site, 2017. `https://github.com/LoicMarechal/libMeshb`.

[13] J. Ahrens, B. Geveci, C. Law, ParaView: An End-User Tool for Large Data Visualization, Elsevier, Washington, D.C., 2005.

[14] T. Michal, J. Krakos, Anisotropic Mesh Adaptation Through Edge Primitive Operations, AIAA Paper 2012–159, 2012.

[15] A. Loseille, F. Alauzet, Continuous mesh framework part i: Well-posed continuous interpolation error, SIAM Journal on Numerical Analysis 49 (2011) 38–60.

[16] M. A. Park, D. L. Darmofal, Parallel Anisotropic Tetrahedral Adaptation, AIAA Paper 2008–917, 2008.

[17] F. Alauzet, A changing-topology moving mesh technique for large displacements, Engineering with Computers 30 (2014) 175–200.

[18] D. A. Ibanez, Conformal Mesh Adaptation on Heterogeneous Supercomputers, Ph.D. thesis, Rensselaer Polytechnic Institute, 2016.

[19] D. Ibanez, M. Shephard, Mesh adaptation for moving objects on shared memory hardware, in: 25th International Meshing Roundtable, Sandia National Laboratories, 2016, pp. 1–5.

[20] D. Ibanez, Omega_h GitHub site, 2017. `https://github.com/ibaned/omega_h`.

[21] G. Gorman, Pragmatic GitHub site, 2017. `https://meshadaptation.github.io`.

[22] G. J. Gorman, G. Rokos, J. Southern, P. H. J. Kelly, Thread-parallel anisotropic mesh adaptation, in: New Challenges in Grid Generation and Adaptivity for Scientific Computing, Springer, 2015, pp. 113–137.

[23] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, H. Zhang, H. Zhang, PETSc Users Manual, Technical Report ANL-95/11 - Revision 3.7, Argonne National Laboratory, 2016. URL: `http://www.mcs.anl.gov/petsc`.

[24] N. Barral, M. G. Knepley, M. Lange, M. D. Piggott, G. J. Gorman, Anisotropic mesh adaptation in Firedrake with PETSc DMPlex, in: 25th International Meshing Roundtable, Washington D. C., VA, USA, 2016.

[25] Y. V. Vasilevskii, K. Lipnikov, An adaptive algorithm for quasioptimal mesh generation, Computational mathematics and mathematical physics 39 (1999) 1468–1486.

[26] A. Loseille, R. Löhner, Anisotropic Adaptive Simulations in Aerodynamics, AIAA Paper 2010–169, 2011.

[27] A. Loseille, Chapter 10 - unstructured mesh generation and adaptation, in: R. Abgrall, C.-W. Shu (Eds.), Handbook of Numerical Methods for Hyperbolic Problems: Applied and Modern Issues, volume 18 of *Handbook of Numerical Analysis*, Elsevier, 2017, pp. 263–302. doi:http://dx.doi.org/10.1016/bs.hna.2016.10.004.

[28] A. Loseille, V. Menier, Serial and parallel mesh modification through a unique cavity-based primitive, 22nd International Meshing Roundtable, Sandia National Laboratories, Springer International Publishing, 2014, pp. 541–558. doi:10.1007/978-3-319-02335-9_30.

[29] A. Loseille, V. Menier, F. Alauzet, Parallel generation of large-size adapted meshes, in: Procedia Engineering, 24th International Meshing Roundtable, Sandia National Laboratories, 2015, pp. 57–69. doi:10.1016/j.proeng.2015.10.122.

[30] A. Loseille, R. Löhner, Robust boundary layer mesh generation, 21st International Meshing Roundtable, Sandia National Laboratories, Springer Berlin Heidelberg, 2013, pp. 493–511. doi:10.1007/978-3-642-33573-0_29.

[31] A. Loseille, Metric-orthogonal anisotropic mesh generation, in: Procedia Engineering, 23rd International Meshing Roundtable, Sandia National Laboratories, 2014, pp. 403–415. doi:10.1016/j.proeng.2014.10.400.

[32] F. Alauzet, Size gradation control of anisotropic meshes, Finite Elements in Analysis and Design 46 (2010) 181–202.

[33] H. Borouchaki, F. Hecht, P. J. Frey, Mesh gradation control, International Journal for Numerical Methods in Engineering 43 (1998) 1143–1165.

[34] S. Z. Pirzadeh, Three-dimensional unstructured viscous grids by the advancing-layers method, AIAA Journal 34 (1996) 43–49.

[35] M. A. Park, J.-R. Carlson, Turbulent Output-Based Anisotropic Adaptation, AIAA Paper 2010–168, 2010.

[36] N. Jansson, J. Hoffman, J. Jansson, Framework for massively parallel adaptive finite element computational fluid dynamics on tetrahedral meshes, SIAM Journal on Scientific Computing 34 (2012) C24–C41.