A Comparison of Tetrahedral Mesh Improvement Techniques

Lori A. Freitag and Carl Ollivier-Gooch

Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, Illinois 60439
{freitag,gooch}@mcs.anl.gov

Abstract. Automatic mesh generation and adaptive refinement methods for complex three-dimensional domains have proven to be very successful tools for the efficient solution of complex applications problems. These methods can, however, produce poorly shaped elements that cause the numerical solution to be less accurate and more difficult to compute. Fortunately, the shape of the elements can be improved through several mechanisms, including face-swapping techniques that change local connectivity and optimization-based mesh smoothing methods that adjust grid point location. We consider several criteria for each of these two methods and compare the quality of several meshes obtained by using different combinations of swapping and smoothing. Computational experiments show that swapping is critical to the improvement of general mesh quality and that optimization-based smoothing is highly effective in eliminating very small and very large angles. The highest quality meshes are obtained by using a combination of swapping and smoothing techniques.

Keywords. Mesh Improvement, Mesh Smoothing

1. Introduction

The use of finite element and unstructured finite volume solution methods is increasingly common for application problems in science and engineering. Regardless of the particular solution technique employed, the computational domain must be decomposed into simple geometric elements. This decomposition can be accomplished by using available automatic mesh generation tools. Unfortunately, meshes generated in this way can contain poorly shaped or distorted elements, which cause numerical difficulties during the solution process. For example, we know that as element dihedral angles become too large, the discretization error in the finite element solution increases [2]; and as angles become too small, the condition number of the element matrix increases [9]. Thus, for meshes with highly distorted elements, the solution is both less accurate and more difficult to compute. This problem is more severe in three dimensions than in two dimensions, because tetrahedra can be distorted to poor quality in more ways than triangles can. Compared with triangular meshes, tetrahedral meshes tend to have a larger proportion of poor quality elements and to have elements that are more severely distorted.

The algorithms for unstructured mesh improvement fall into three basic categories:

1. point insertion/deletion to refine or coarsen a mesh [3], [14], [16],
2. local reconnection or face swapping to change mesh topology for a given set of vertices [6], [10], [11], and
3. mesh smoothing to relocate grid points to improve mesh quality without changing mesh topology [1], [4], [15].

In this article, we follow a two-pronged approach to improve the quality of tetrahedral meshes. We start by swapping mesh faces to improve the connectivity and follow with mesh smoothing to adjust grid point position. Face swapping techniques are widely used, and we give only a brief overview of the methods used. The most common approaches to mesh smoothing are variants on Laplacian smoothing [7]. While

these smoothers are often effective, they operate heuristically with no effort to locate points specifically to improve some quality measure. In this article, we present an optimization-based smoothing algorithm for tetrahedral meshes. This algorithm is an extension of a highly effective and efficient two-dimensional smoothing algorithm [8].

The remainder of the article is organized as follows. In Section 2, we briefly summarize face-swapping techniques. In Section 3, we describe a mesh smoothing algorithm using local optimization. We then present the results of numerical experiments on several test meshes. Mesh quality improvement by face swapping, vertex smoothing, and combinations of swapping and smoothing is presented. Finally, in Section 5, we offer concluding remarks and directions for future research.

2. Face Swapping

Face swapping changes the local connectivity of a simplicial mesh to improve mesh quality. Each interior face in a tetrahedral mesh separates two tetrahedra made up of a total of five points. A large number of nonoverlapping tetrahedral configurations are possible with these five points, but only two can be legally reconnected, or swapped. These two cases are shown in Figure 1. On the left is a case in which either two or three tetrahedra can be used to fill the convex hull of a set of five points. Switching from two to three tetrahedra requires the addition of an edge interior to the convex hull. On the right of the figure is a configuration in which two tetrahedra can be exchanged for two different ones. The shaded faces in the figure are coplanar, and swapping exchanges the diagonal of the coplanar quadrilateral. The two coplanar faces must either be boundary faces or be backed by another pair of tetrahedra that can be swapped two for two. Otherwise, the new edge created by the two for two swap will not be conformal.

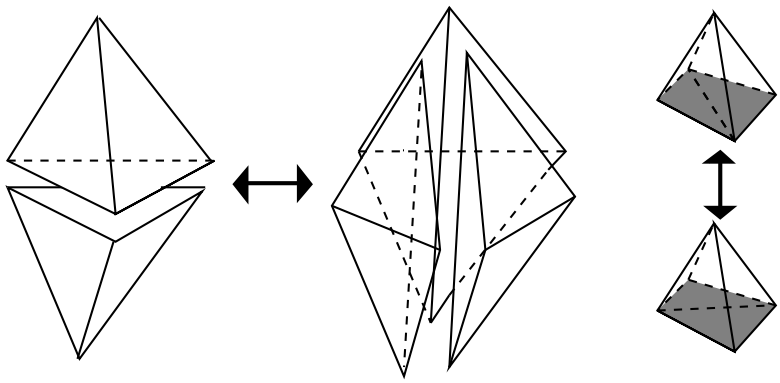

Figure 1: Swappable configurations of five points in three dimensions

Because each reconfigurable case has only two valid configurations, a quick comparison to find the one with the higher quality is possible. If the higher-quality configuration is not already present, reconnection is performed to obtain it. In the case of configurations of equal quality, we select the two-tet configuration when choosing between two and three tet configurations, and we choose not to swap in the two-for-two reconfiguration case.

We use two geometric quality measures to determine whether to locally reconnect a tetrahedral mesh: the minmax dihedral angle criterion and the in-sphere criterion. The minmax dihedral angle criterion chooses the configuration that minimizes the maximum dihedral angle of the tetrahedra formed by the five points in the two tets incident on a face. The in-sphere criterion selects the configuration in which no tetrahedron formed by four of the five points contains the other point in its circumsphere. This leads to a locally Delaunay tetrahedralization in the sense that there is no face in the mesh with incident cells violating the in-sphere criterion that are reconfigurable. For either criterion, however, the optimum reached by this face-swapping algorithm will probably be local rather than global. Recent work by Joe [11] describes a more advanced technique for improving mesh quality by local transformations. This approach notwithstanding, however, it is not known whether the global optimum can be reached by *any* series of local transformations.

3. An Optimization Approach to Mesh Smoothing

Perhaps the most commonly used mesh-smoothing technique is a local algorithm called Laplacian smoothing [7], [13]. This technique adjusts the location of each grid point to the arithmetic mean of its incident vertices so that

$$x_{free} = \frac{\sum_{i \in V} x_i}{|V|}, \quad y_{free} = \frac{\sum_{i \in V} y_i}{|V|}, \quad z_{free} = \frac{\sum_{i \in V} z_i}{|V|}, \tag{1}$$

where V is the set of incident vertices and x, y, and z are the spatial coordinates of each vertex. This method is computationally inexpensive, but it does not provide any mechanisms that guarantee improvement in element quality. In fact, it is possible to produce an invalid mesh containing elements that are inverted or have negative volume.

Freitag et. al. proposed a low-cost, optimization-based alternative to Laplacian smoothing that guarantees valid elements in the final mesh [8]. Several results were given that demonstrated the effectiveness of this method compared with Laplacian smoothing for two-dimensional, triangular meshes. Like Laplacian smoothing, the optimization algorithm is local and uses the union of elements that are adjacent to the free vertex as the solution space. Thus, it can be used as the core of an efficient parallel algorithm. They presented a P-RAM computational model for parallel implementation based on coloring heuristics. This model resulted in correct parallel execution and a low run-time bound, and experimental data showed very good scalable performance on 1 to 64 processors on the IBM SP supercomputer.

In this section we extend this algorithm to three-dimensional tetrahedral meshes and note that the algorithm is useful for hexahedral meshes as well. A parallel algorithm analogous to the two-dimensional algorithm has been developed for the three-dimensional case, but we do not focus on that aspect of our work here. In this article we describe the formulation of the optimization method and give some useful measures of mesh quality for tetrahedral meshes. The same formulation applies for hexahedral meshes, but different mesh quality measures must be used. As in the two-dimensional case, we formulate the problem using analytic expressions for local mesh quality written in terms of free vertex position. Typical measures for three-dimensional tetrahedral meshes that have an analytic expression include measures of the dihedral angles, measures of the solid angles, and element aspect ratios. Any combination of these can be used within the framework of the optimization method presented here. Our algorithm seeks to maximize the minimum value of the mesh quality measure; minimizing the maximum value of the quality measure can be done by negating the function value. We require that function and gradient evaluations dependent on free vertex position be provided by the user.

The optimization algorithm for each local subproblem is similar to a minimax technique used to solve circuit design problems [5]. We briefly review the formulation of the problem here and refer interested readers to a more complete description in our previous paper [8]. To facilitate the discussion of problem formulation, we first introduce some useful notation:

- $\mathbf{x}$: the position of the free vertex
- $f_i(\mathbf{x})$: an analytic function for a mesh quality measure that in general is nonlinear. For example, if we consider maximizing the minimum dihedral angle of the mesh, each tetrahedron will have six function values for each location of the point $\mathbf{x}$. Let the entire set of function values, f_i, be $\mathcal{S}$.
- $\mathbf{g}_i(\mathbf{x})$: the analytic gradient of the mesh quality measure corresponding to $f_i(\mathbf{x})$
- $\mathcal{A}$: the set of functions that achieve a minimum at point $\mathbf{x}$ (the active set)
- $\mathbf{x}^*$, $\mathcal{A}^*$: the optimal solution and the active set at $\mathbf{x}^*$

As the location of $\mathbf{x}$ changes in the solution space, the minimum function value in the corresponding submesh is given by the composite function

$$\phi(\mathbf{x}) \;=\; \min_{i \in S} \; f_i(\mathbf{x}). \tag{2}$$

We illustrate the character of this function by showing a one-dimensional slice through a typical function ϕ in Figure 2. Note that each $f_i(\mathbf{x})$ is a smooth, continuously differentiable function and that multiple

function values can obtain the minimum value. Hence, the composite function $\phi(\mathbf{x})$ has discontinuous partial derivatives where two or more of the functions f_i obtain the minimum value; that is, where the active set $\mathcal{A}$ changes.

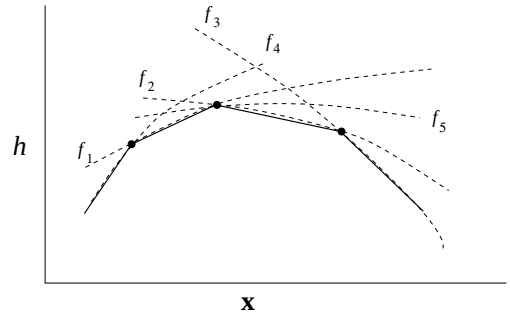

Figure 2: A one dimensional slice through the nonsmooth function $\phi(x)$

.

We solve the nonsmooth optimization problem (2) using an analogue of the steepest descent method for smooth functions. The search direction $\bar{\mathbf{g}}$ at each step is computed by solving the quadratic programming problem

$$\min \ \bar{\mathbf{g}}^T \bar{\mathbf{g}} \quad \text{where} \quad \bar{\mathbf{g}} = \sum_{i \in \mathcal{A}} \beta_i \mathbf{g}_i(\mathbf{x})$$

$$\text{subject to} \ \sum_{i \in \mathcal{A}} \beta_i \ = \ 1, \quad \beta_i \geq 0$$

for the β_i. This gives the direction of steepest descent from all possible convex linear combinations of the gradients in the active set at $\mathbf{x}$. The line search subproblem along $\bar{\mathbf{g}}$ is solved by finding the linear approximation of each $f_i(\mathbf{x})$ given by the first-order Taylor series approximation. Since this information is available for each f_i, we can predict the points at which the active set $\mathcal{A}$ will change. These points are given by the intersection of each linear approximation with the projection the current active function in the search direction. The distance to the nearest intersection point from the current location gives the initial step length, α. The initial step is accepted if the actual improvement exceeds 90 percent of the estimated improvement or the subsequent step results in a smaller function improvement. Otherwise α is halved recursively until a step is accepted or α falls below some minimum step length tolerance.

The optimization process is terminated if one of the following conditions apply: (1) the step size falls below the minimum step length with no improvement obtained; (2) the maximum number of iterations is exceeded; (3) the achieved improvement of any given step is less than some user-defined tolerance; or (4) the Kuhn-Tucker conditions of nonlinear programming

$$\sum_{i \in \mathcal{A}^*} \lambda_i \mathbf{g}_i(\mathbf{x}^*) = 0$$

$$\sum_{i \in \mathcal{A}^*} \lambda_i = 1, \quad \lambda_i \geq 0, \quad i \in \mathcal{A}^*$$

are satisfied indicating that we have found a local maximum $\mathbf{x}^*$ [5].

4. Computational Experiments

This paper presents results for five test cases: two randomly generated meshes and three meshes generated for application problems. For each test mesh, a standard starting mesh was generated, and all comparison cases began with that same mesh. The random meshes were each generated in a cube with points incrementally inserted at random in the interior. Each point was connected to the vertices of the tetrahedron

containing it, with points near an existing face or edge in the tetrahedralization projected onto that face or edge. No swapping or smoothing was done as these initial meshes were generated, and mesh quality was correspondingly poor at the outset. The first case, `rand1`, has 1086 points approximately equally distributed through the domain and 5104 tetrahedra. The second random mesh, `rand2` has 5086 points clustered at the center of the cube by selecting random numbers from a Gaussian deviate and 25704 tetrahedra.

The third and fourth test meshes were generated in the interior of a tangentially-fired (t-fired) industrial boiler and a tire incinerator, respectively. Interior points were inserted at the circumcenter of cells that were larger than appropriate, based on an automatically computed local length scale, or that had a large dihedral angle and had a volume larger than a user-defined tolerance. After the insertion of each point, nearby faces were swapped by using the in-sphere criterion to improve local mesh connectivity. After all points were inserted, all faces were swapped by using the minmax dihedral angle criterion. The t-fired mesh has 7265 vertices and 37785 tetrahedra, and the tire incinerator has 2570 vertices and 11098 tetrahedra. Because these meshes were generated more sensibly than the random meshes, its initial quality is much better than in the random cases.

The final test case is a mesh generated around the ONERA M6 wing attached to a flat wall. This is a standard geometry for testing three-dimensional compressible flow solution algorithms. This particular mesh is somewhat coarse, having 6,000 vertices and 31,978 tetrahedra. Hence, the initial mesh quality is somewhat poor, especially near the junction of the wing and the wall.

For each test case, we present results for mesh quality using dihedral angles as a quality measure. The maximum and minimum dihedral angles over the entire mesh are given as an indication of how poor the worst elements are. To give quantitative information about the number of poor tetrahedra, we also give the percentage of dihedral angles falling below 6, 12, and 18 degrees and above 162, 168, and 174 degrees.

4.1 Improvement Using Mesh Swapping Techniques Only

The first experiment compares the effectiveness of several swapping strategies in improving mesh quality for the random meshes. Tables 1 and 2 show mesh quality results for the initial random meshes and the same meshes following face swapping. The results of employing four face-swapping strategies are shown: the in-sphere and minmax dihedral criteria separately, in-sphere followed by minmax, and minmax followed by in-sphere. For each mesh, swapping using the in-sphere criterion followed by the minmax dihedral criterion results in meshes with far fewer dihedral angles near the unwanted extremes of 0 and 180 degrees. However, in each case, the in-sphere criterion, whether used alone or in conjunction with the minmax angle criterion, introduces some tetrahedra with extremely poor angles into the mesh. Consequently, use of the minmax angle criterion alone gives the best extremal angles.

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Initial	0.32°	178.97°	1.41	4.90	9.86	2.40	1.08	0.24
In-sphere	$3.6 \cdot 10^{-6}$°	180.00°	1.07	3.17	6.35	1.30	0.70	0.29
Minmax angle	0.54°	178.97°	0.76	3.20	7.40	1.21	0.46	0.11
In-sphere, then minmax angle	$3.6 \cdot 10^{-6}$°	180.00°	0.45	1.48	3.28	0.58	0.30	0.11
Minmax angle, then In-sphere	$3.6 \cdot 10^{-6}$°	180.00°	1.14	3.23	6.36	1.31	0.74	0.31

Table 1: Mesh quality improvement for `rand1` with swapping

4.2 Improvement Using Mesh Smoothing Techniques Only

Our baseline smoothing technique is Laplacian smoothing. Our Laplacian smoothing algorithm moves each vertex to the average of the location of its neighbors, provided that this point location does not result in an invalid mesh. Also, we have implemented a "smart" variant on Laplacian smoothing which also requires

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Initial	0.10°	179.84°	2.57	8.33	14.77	4.24	2.04	0.51
In-sphere	$3.5 \cdot 10^{-6}$°	180.00°	1.49	3.96	7.39	1.73	1.00	0.38
Minmax angle	0.57°	179.20°	1.51	5.83	11.52	2.51	1.07	0.21
In-sphere then minmax angle	$6.0 \cdot 10^{-6}$°	180.00°	0.60	1.82	3.78	0.77	0.43	0.15
Minmax angle then In-sphere	$3.5 \cdot 10^{-6}$°	180.00°	1.38	3.81	7.21	1.64	0.94	0.35

Table 2: Mesh quality improvement for `rand2` with swapping.

that some local mesh quality measure be improved before accepting a change in vertex location. Any local quality criterion suitable for use with the optimization-based smoothing can be used in this context.

For optimization-based smoothing, we present results for five different objective functions:

1. Maximize the minimum dihedral angle (maxmin angle)
2. Minimize the maximum dihedral angle (minmax angle)
3. Maximize the minimum cosine of the dihedral angles (maxmin cosine)
4. Minimize the maximum cosine of the dihedral angles (minmax cosine)
5. Maximize the minimum sine of the dihedral angles (maxmin sine)

We expect nearly identical results, though not necessarily identical convergence behavior, from two pairs of these measures:

$$\begin{aligned} \text{maxmin angle} &\approx \text{minmax cosine} \\ \text{minmax angle} &\approx \text{maxmin cosine.} \end{aligned}$$

Tables 3 and 4 show the results of smoothing the random meshes using Laplacian smoothing, smart Laplacian smoothing for two of the five criteria given, and optimization-based smoothing for each of the five criteria. Results for the other smart Laplacian approaches are not shown because they are identical to one of the two smart Laplacian results presented.

The improvement in mesh quality is not as pronounced for smoothing as for swapping because the connectivity is too irregular to allow a truly high-quality mesh. Nevertheless, all the optimization-based smoothing criteria improve mesh quality significantly, especially in the sense of reducing the number of elements with extremely poor dihedral angles. The Laplacian smoother does a poor job of eliminating very bad angles. The smart Laplacian smoothers perform better in this respect, but still are significantly worse than the optimization-based smoothers. Optimization criteria that seek only to force all dihedral angles away from 180 degrees (minmax angle and maxmin cosine) are unsuccessful in eliminating small dihedral angles, while criteria that force dihedral angles away from 0 degrees (maxmin angle and minmax cosine) succeed in also eliminating large dihedral angles. This difference can be important in practice as both large and small angles can affect the quality of the final application solution. Note also that the pairs of smoothing criteria expected to perform comparably behave similarly, with one exception. The minmax cosine criterion does not improve the extremal angles as much as its analog, the maxmin angle criterion. The flatness of the cosine near 0 and 180 degrees prevents rapid quality improvement when moving points in such tetrahedra, and the optimization code concludes that improvement is too slow to be fruitful. Finally, the maxmin sine criterion, which should force dihedral angles away from both 0 and 180 degrees, is very successful in removing dihedral angles at both extremes.

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Lap: No constraint	0.054°	179.93°	1.32	4.02	7.82	1.92	1.02	0.39
Lap: Maxmin angle	1.18°	177.43°	0.71	3.23	7.28	1.55	0.58	0.14
Lap: Minmax angle	0.67°	177.43°	0.73	3.44	7.55	1.44	0.54	0.091
Opt: Maxmin angle	4.79°	175.59°	0.11	1.23	6.21	0.71	0.21	0.0065
Opt: Minmax angle	$5.37 \cdot 10^{-4}$°	172.75°	2.71	5.54	9.09	0.16	0.043	0.00
Opt: Maxmin cosine	$6.34 \cdot 10^{-3}$°	172.56°	2.29	4.85	8.88	0.16	0.020	0.00
Opt: Minmax cosine	1.71°	176.21°	0.11	1.31	6.49	0.75	0.19	0.016
Opt: Maxmin sine	4.20°	175.73°	0.08	1.05	6.09	0.60	0.11	0.0033

Table 3: Mesh quality improvement for `rand1` with smoothing

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Lap: No constraint	0.0026°	179.996°	2.45	6.90	12.21	3.46	1.84	0.63
Lap: Maxmin angle	0.64°	178.76°	1.58	6.31	12.35	3.03	1.35	0.26
Lap: Minmax angle	0.51°	178.83°	1.71	6.39	12.35	2.95	1.24	0.23
Opt: Maxmin angle	2.64°	178.35°	0.25	4.47	11.93	1.98	0.59	0.053
Opt: Minmax angle	$5.59 \cdot 10^{-5}$°	174.53°	3.62	7.69	12.93	1.11	0.21	0.0006
Opt: Maxmin cosine	$1.25 \cdot 10^{-4}$°	175.69°	3.30	7.25	12.60	1.00	0.18	0.0045
Opt: Minmax cosine	0.10°	179.84°	0.49	4.68	11.93	2.16	0.73	0.12
Opt: Maxmin sine	2.58°	177.16°	0.27	4.45	11.75	1.97	0.58	0.019

Table 4: Mesh quality improvement for `rand2` with smoothing

4.3 Improvement Using the Combined Swapping/Smoothing Approach

Next we show that the gains in mesh quality from swapping and smoothing can be made cumulatively. Each swapping technique results in a different distribution of tetrahedron shapes. The minmax dihedral angle swapping results in a mesh a good distribution of angles near 0° and 180° but does not improve the roundness of the tetrahedra. In contrast, the in-sphere swapping technique followed by the minmax dihedral angle swapping results in angles very close to 0° and 180° but with an improved overall shape distribution. To determine which of these two swapping criterion is best to use in a combined approach, we compare the results of swapping followed by six passes of optimization-based smoothing using both the the maxmin angle and maxmin sine criteria for the first random mesh. Table 5 shows that both smoothing criteria effectively eliminate the very poor tetrahedra in the in-sphere–minmax swapping case, but are not able to improve the distribution of distorted elements if minmax swapping is used alone.

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Minmax only Maxmin angle	2.85°	178.66°	0.13	2.84	9.49	1.40	0.43	0.048
In-sphere, minmax Maxmin angle	7.46°	175.95°	0.00	0.061	0.78	0.090	0.020	0.0011
Minmax only Maxmin sine	3.05°	176.11°	0.080	2.74	9.46	1.33	0.30	0.0050
In-sphere, minmax Maxmin sine	7.50°	170.09°	0.00	0.065	0.80	0.079	0.0040	0.00

Table 5: Comparison of the effectiveness of smoothing for two different swapping options (mesh `rand1`)

Tables 6 and 7 show the results for the two random meshes of using the best swapping combination—in-sphere swapping, then minmax dihedral angle swapping—followed by each of the eight smoothing options discussed in the preceding section. The distribution of dihedral angles for each random mesh improves significantly regardless of the choice of smoothing criterion. As was the case with smoothing used alone, all Laplacian smoothers fail to eliminate poorly shaped elements from the mesh. Again we see that the criteria for optimization-based smoothing that seek only to remove large angles from the mesh do not succeed in eliminating small angles; however the criteria that eliminate small angles also succeed in eliminating large angles. Maximizing the minimum sine of the dihedral angles is the most successful smoothing criterion for eliminating poor dihedral angles at both extremes.

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Lap: No constraint	0.12°	179.77°	0.26	0.60	1.25	0.29	0.18	0.085
Lap: Maxmin angle	0.47°	179.22°	0.14	0.35	1.02	0.18	0.088	0.048
Lap: Minmax angle	0.47°	179.22°	0.19	0.54	1.46	0.19	0.12	0.048
Opt: Maxmin angle	13.72°	169.07°	0.00	0.00	0.28	0.020	0.0028	0.00
Opt: Minmax angle	$3.9 \cdot 10^{-3}$°	167.93°	1.28	2.74	4.73	0.031	0.00	0.00
Opt: Maxmin cosine	0.026°	168.73°	1.40	2.78	4.79	0.017	0.0056	0.00
Opt: Minmax cosine	$3.6 \cdot 10^{-6}$°	180.00°	.011	.042	0.42	0.045	0.0085	0.0056
Opt: Maxmin sine	11.77°	163.33°	0.00	0.0028	0.24	0.0085	0.00	0.00

Table 6: Mesh quality improvement for `rand1` with both swapping and smoothing

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Lap: No constraint	$3.4 \cdot 10^{-4}$°	180.00°	0.40	0.97	1.89	0.48	0.28	0.13
Lap: Maxmin angle	0.22°	179.43°	0.18	0.72	1.75	0.34	0.16	0.042
Lap: Minmax angle	$1.4 \cdot 10^{-4}$°	180.00	0.22	0.84	2.04	0.33	0.15	0.042
Opt: Maxmin angle	7.46°	175.95°	0.00	0.061	0.78	0.090	0.020	0.0011
Opt: Minmax angle	$1.19 \cdot 10^{-3}$°	171.62°	1.56	3.13	5.38	0.031	0.0028	0.00
Opt: Maxmin cosine	$4.80 \cdot 10^{-4}$°	175.60°	1.56	3.10	5.38	0.034	0.020	0.0023
Opt: Minmax cosine	3.65°	175.39°	0.0045	0.095	0.83	0.10	0.030	0.0028
Opt: Maxmin sine	7.50°	170.09°	0.00	0.065	0.80	0.079	0.0040	0.00

Table 7: Mesh quality improvement for `rand2` with both swapping and smoothing

Passes	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
0	$3.62 \cdot 10^{-6}$°	180.00°	0.45	1.48	3.27	0.58	0.30	0.11
1	3.58°	174.80°	0.034	0.49	1.61	0.24	0.051	0.0056
2	5.34°	171.27°	0.0085	0.00	0.98	0.18	0.017	0.098
3	8.01°	169.30°	0.00	0.073	0.68	0.045	0.0028	0.00
4	9.70°	166.97°	0.00	0.045	0.50	0.023	0.00	0.00
5	10.40°	164.50°	0.00	0.014	0.34	0.020	0.00	0.00
6	11.77°	163.33°	0.00	0.0028	0.24	0.0085	0.00	0.00

Table 8: Effect of the number of optimization passes on mesh improvement (`rand1` with maxmin sine smoothing)

An important question in any local smoothing algorithm is the number of smoothing passes required to improve the mesh to the point where further improvement is negligible. Tables 8 and 9 show the effect of

Passes	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
0	$6.04 \cdot 10^{-6}$°	180.00°	0.60	1.82	3.78	0.77	0.43	0.16
1	0.67°	178.88°	0.13	0.70	2.26	0.33	0.12	0.026
2	1.10°	179.01°	0.031	0.32	1.50	0.18	0.051	0.0085
3	3.32°	177.31°	0.0096	0.19	1.11	0.15	0.039	0.0011
4	5.46°	175.99°	0.0017	0.12	0.94	0.12	0.029	0.0017
5	6.66°	176.38°	0.00	0.087	0.83	0.10	0.024	0.0011
6	7.46°	175.95°	0.00	0.061	0.78	0.090	0.020	0.0011

Table 9: Effect of the number of optimization passes on mesh improvement (`rand2` with minmax angle smoothing)

Case	Swap Time (sec)	Smoothing Time (sec)	Smoothing Calls	Time per Call (msec)	Min Dihed	Max Dihed
Swap only	5.19	—	—	—	0.57°	179.20°
Lap: No constraint	—	6.54	14382	0.46	0.014°	179.98°
Lap: Maxmin sine	—	37.2	14382	2.59	0.63°	178.83°
Opt: Maxmin sine	—	244	14382	17.0	1.91°	177.69°
Swap + Lap: No constraint	5.19	6.54	14382	0.46	0.0033°	179.99°
Swap + Lap: Maxmin sine	5.19	37.2	14382	2.59	0.63°	178.76°
Swap + Opt: Maxmin sine	5.19	258	14382	17.9	1.98°	177.85°

Table 10: Sample times for mesh improvement (mesh `rand2`)

various numbers of smoothing passes with the maxmin sine criterion on `rand1` and with the maxmin angle criterion on `rand2`. In both cases swapping was used before smoothing. In each case, mesh quality improves only negligibly after the fourth smoothing pass.

We conclude this section with a word about computational efficiency of the optimization-based smoothing approach. Table 10 compares timings for mesh improvement using swapping, smoothing and a combination for `rand2` on a 110 MHz SPARC 5. The difference in timings for smoothing with and without swapping is due to differences in the number of optimization steps required for the different meshes. The time spent swapping is small compared with smart Laplacian or optimization-based smoothing and is therefore certainly a good investment. For these examples, optimization-based smoothing takes 7 times longer than smart Laplacian smoothing and 38 times longer than simple Laplacian smoothing. Further work is needed to quantify the gains, if any, in solution speed and accuracy.

4.4 Improvement of Application Meshes

Because the application meshes have already been improved by swapping, results are presented only for smoothing. Table 11 shows the effect of smoothing on the dihedral angle distribution for the t-fired mesh using each smoothing criterion. As with the random meshes, Laplacian smoothing does little to remove extremely poor dihedral angles. The use of measures that concentrate on decreasing the maximum dihedral angles (minmax angle and maxmin cosine) has the effect of allowing relatively large numbers of small dihedral angles, although the improvement of elements with large dihedral angles is quite striking. The minmax cosine criterion again fails to improve extremal angles as much as its cousin, maxmin angle. The difference between maxmin angle and maxmin sine is fairly small, with maxmin angle giving better extremal angles and maxmin sine giving a somewhat better angle distribution.

The effectiveness of various numbers of smoothing passes on dihedral angle distribution for this case is shown in Table 12 for the maxmin sine criterion. Two optimization passes give nearly all the improvement that is possible. For comparison, the two-dimensional version of this algorithm typically requires one to two optimization passes to improve meshes of reasonable quality [8]. The reason fewer optimization passes are

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
Initial	0.259°	179.632°	0.13	0.45	0.92	0.21	0.10	0.026
Lap: No constraint	0.26°	179.63°	0.055	0.13	0.26	0.073	0.044	0.020
Lap: Maxmin angle	0.26°	179.63°	0.032	0.090	0.18	0.050	0.025	0.0093
Lap: Minmax angle	0.26°	179.63°	0.038	0.11	0.24	0.054	0.028	0.0093
Opt: Maxmin angle	2.07°	177.180°	0.014	0.057	0.099	0.027	0.015	0.0018
Opt: Minmax angle	0.0315°	174.834°	0.32	0.81	1.69	0.035	0.019	0.0049
Opt: Maxmin cosine	0.0070°	177.161°	0.28	0.73	1.59	0.035	0.019	0.0044
Opt: Minmax cosine	0.42°	178.313°	0.021	0.058	0.10	0.032	0.016	0.0044
Opt: Maxmin sine	1.78°	176.671°	0.016	0.066	0.11	0.033	0.018	0.0013

Table 11: Mesh quality improvement for `tfire` with smoothing

Passes	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
0	0.26°	179.63°	0.13	0.45	0.92	0.21	0.10	0.026
1	0.92°	177.33°	0.027	0.094	0.19	0.047	0.024	0.0066
2	1.35°	176.34°	0.022	0.067	0.12	0.035	0.019	0.0040
3	1.59°	176.59°	0.019	0.065	0.11	0.034	0.019	0.0031
4	1.71°	176.85°	0.018	0.063	0.11	0.033	0.018	0.0018
5	1.74°	176.76°	0.018	0.064	0.11	0.033	0.017	0.0018
6	1.78°	176.67°	0.016	0.066	0.11	0.033	0.018	0.0013

Table 12: Effect of the number of optimization passes on mesh improvement (`tfire` with minmax angle smoothing)

required for this case than for the random meshes is that the distribution of points is far better initially, resulting in less overall point movement.

Table 13 shows the improvement in mesh quality achieved for each of the three application meshes. For all three cases, mesh quality is improved significantly. The final mesh quality differs dramatically among the three cases, because of the initial topology and point distribution of the meshes. For example, the M6 wing mesh began with a very large number of poor dihedral angles in adjacent tetrahedra. While smoothing improved many tetrahedra, some could not be improved without making a neighboring cell worse, and so no improvement was made.

This clustering of bad tetrahedra is a common occurrence in our final meshes, with the worst cells often sharing vertices, edges, or even faces. Figures 3 and 4 show surface wireframes for the t-fired boiler and tire incinerator, along with the worst tetrahedra—those with dihedral angles less than 10° or greater than 160°. For the t-fired boiler, these tetrahedra fall primarily into a single clump along an corner of the geometry. Figure 5 shows a closeup of a region around the leading edge of the wing at the wall where there is a

Case	Min Dihed	Max Dihed	% dihedral angles <			% dihedral angles >		
			6°	12°	18°	162°	168°	174°
T-fire boiler before	0.26°	179.63°	0.13	0.45	0.92	0.21	0.10	0.026
T-fire boiler after	1.59°	176.59°	0.019	0.065	0.11	0.034	0.018	0.0018
Tire incinerator before	0.66°	178.88°	0.11	0.54	1.27	0.072	0.035	0.0075
Tire incinerator after	4.34°	174.28°	0.0045	0.014	0.10	0.0060	0.0030	0.0015
ONERA M6 wing before	0.0066°	179.984°	0.78	1.63	2.85	0.57	0.41	0.23
ONERA M6 wing after	0.035°	179.929°	0.28	0.79	1.69	0.25	0.13	0.048

Table 13: Mesh improvement for three application meshes

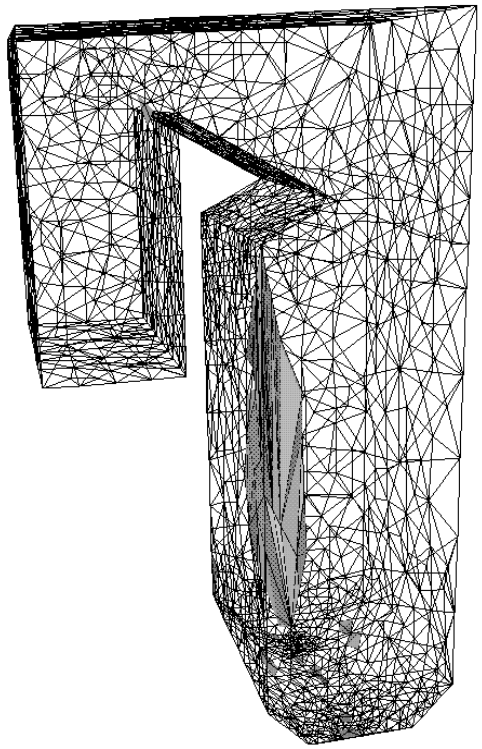

Figure 3: Surface wireframe of tangentially fired boiler mesh with badly shaped tets

concentration of poor-quality tetrahedra. Further work is needed to improve quality in difficult cases such as these in which boundary constraints or clustering prevents the improvement of poorly shaped elements.

5. Conclusions

In this article we compared combinations of mesh swapping and mesh smoothing techniques used to improve the quality of tetrahedral meshes. We showed that each mechanism fails to give high-quality meshes when used individually. That is, all combinations of swapping using in-sphere and minmax quality criteria fail to remove very small and very large angles. Both Laplacian and optimization-based smoothing techniques fail to improve the general distribution of angles because they cannot change local mesh connectivity. However, we showed for five test cases that the cumulative improvement obtained when combining in-sphere and minmax swapping followed by optimization-based smoothing results in very high quality meshes. Of the smoothing criteria considered here, we found that the maxmin sine quality measure was the most effective in eliminating both small and large angles. In addition, we presented evidence that the remaining poor-quality tetrahedra could not be improved by our current methods because these tetrahedra tend to be clustered together. In this situation, swapping fails because local reconnection is not legal, and smoothing fails because improving one tetrahedron reduces the quality of a neighbor.

Several enhancements are being incorporated into the mesh improvement software to increase its effectiveness and efficiency. Our current software uses mesh smoothing to improve the quality of the volume mesh once the surface mesh has been generated. We plan to add surface mesh-smoothing capabilities to the optimization-based algorithm by incorporating additional constraints to bind the free vertex to the boundary surfaces. We are also interested in examining optimization-based smoothing with other measures including aspect ratio and solid angles and in developing smoothing measures appropriate for use on anisotropic meshes. Finally, we intend to investigate the use of more sophisticated local reconnection algorithms, such as that of Joe [11], and various other swapping criteria, including swapping to maximize the minimum sine of dihedral

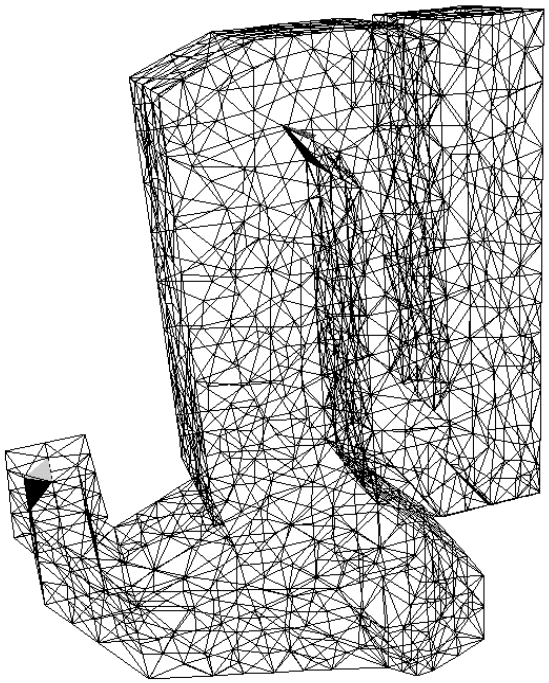

Figure 4: Surface wireframe of tire incinerator mesh with badly shaped tets

angle.

This software is currently being incorporated into the SUMAA3d [12] and GRUMMP projects at Argonne National Laboratory, which will provide an integrated framework for parallel unstructured mesh applications. Therefore we are working to develop parallel algorithms similar to the framework given previously [8] for three-dimensional mesh smoothing and face swapping methods.

Acknowledgments

This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

References

[1] E. Amezua, M. V. Hormaza, A. Hernandez, and M. B. G. Ajuria. A method of the improvement of 3d solid finite-element meshes. *Advances in Engineering Software*, 22:45–53, 1995.

[2] I. Babuska and A. Aziz. On the angle condition in the finite element method. *SIAM Journal on Numerical Analysis*, 13:214–226, 1976.

[3] Randolph E. Bank, Andrew H. Sherman, and Alan Weiser. Refinement algorithms and data structures for regular local mesh refinement. In R. Stepleman et al., editor, *Scientific Computing*, pages 3–17. IMACS/North-Holland Publishing Company, Amsterdam, 1983.

[4] Scott Canann, Michael Stephenson, and Ted Blacker. Optismoothing: An optimization-driven approach to mesh smoothing. *Finite Elements in Analysis and Design*, 13:185–190, 1993.

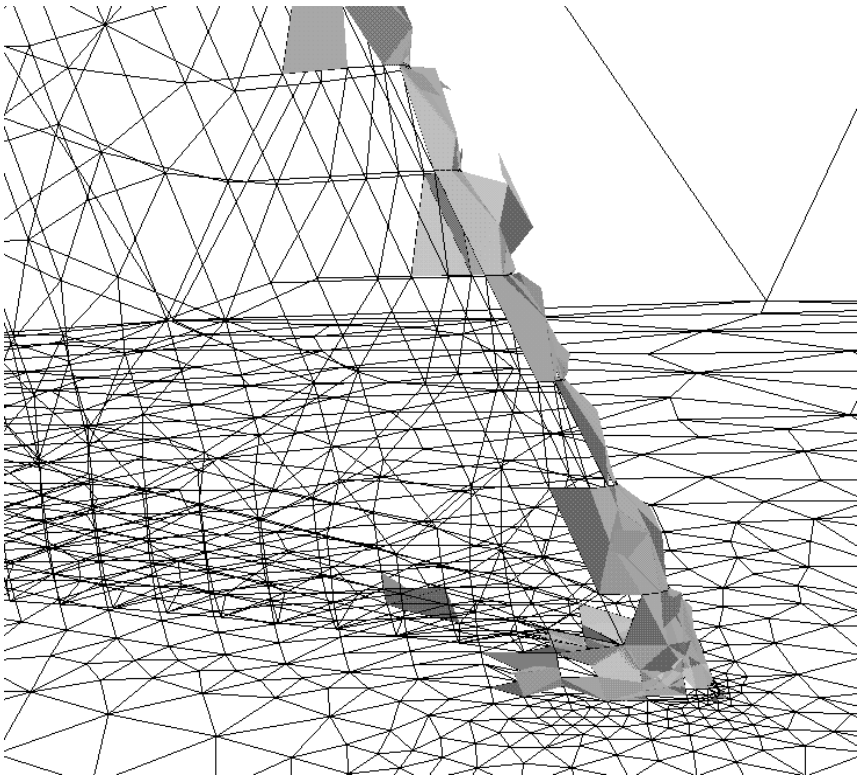

Figure 5: Closeup of leading edge of ONERA M6 wing surface mesh with badly shaped tets.

[5] C. Charalambous and A. Conn. An efficient method to solve the minimax problem directly. *SIAM Journal of Numerical Analysis*, 15(1):162–187, 1978.

[6] H. Edelsbrunner and N. Shah. Incremental topological flipping works for regular triangulations. In *Proceedings of the 8th ACM Symposium on Computational Geometry*, pages 43–52, 1992.

[7] David A Field. Laplacian smoothing and Delaunay triangulations. *Communications and Applied Numerical Methods*, 4:709–712, 1988.

[8] Lori A. Freitag, Mark T. Jones, and Paul E. Plassmann. An efficient parallel algorithm for mesh smoothing. In *Proceedings of the Fourth International Meshing Roundtable*, pages 47–58, Albuquerque, New Mexico, 1995. Sandia National Laboratories.

[9] I Fried. Condition of finite element matrices generated from nonuniform meshes. *AIAA Journal*, 10:219–221, 1972.

[10] Barry Joe. Three-dimensional triangulations from local transformations. *SIAM Journal on Scientific and Statistical Computing*, 10:718–741, 1989.

[11] Barry Joe. Construction of three-dimensional improved quality triangulations using local transformations. *SIAM Journal on Scientific Computing*, 16:1292–1307, 1995.

[12] Mark T. Jones and Paul E. Plassmann. Computational results for parallel unstructured mesh computations. *Computing Systems in Engineering*, 5(4-6):297–309, 1994.

[13] S. H. Lo. A new mesh generation scheme for arbitrary planar domains. *International Journal for Numerical Methods in Engineering*, 21:1403–1426, 1985.

[14] Carl F. Ollivier-Gooch. Multigrid acceleration of an upwind Euler solver on unstructured meshes. *AIAA Journal*, 33(10):1822–1827, 1995.

[15] V. N. Parthasarathy and Srinivas Kodiyalam. A constrained optimization approach to finite element mesh smoothing. *Finite Elements in Analysis and Design*, 9:309–320, 1991.

[16] M. Rivara. Mesh refinement processes based on the generalized bisection of simplices. *SIAM Journal on Numerical Analysis*, 21:604–613, 1984.