INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2003; **58**:319–332 (DOI: 10.1002/nme.768)

A method for hexahedral mesh shape optimization[‡]

Patrick M. Knupp[*,†]

Parallel Computing Sciences Department, Sandia National Laboratories, M/S 0847, P.O. Box 5800, Albuquerque, NM 87185-0847, U.S.A.

SUMMARY

Methods for improving the quality of all-hexahedral unstructured meshes by node-movement strategies have, until recently, been lacking. Laplacian smoothing, while easily implemented and well-known, fails to guarantee improvement of mesh quality and may result in inverted elements where none existed before. A method for improving unstructured hexahedral mesh shape-quality that guarantees untangled elements is proposed. The method is based on optimization of an objective function built from the quality of individual hexahedral elements. The shape-quality measure for hexahedral elements is based on the condition number of a set of Jacobian metrics associated with the element. The theory of the condition number quality metric and of the objective function are reviewed. A numerical optimization procedure to find the improved-quality mesh is described. The purpose of this paper is to demonstrate the robustness of the method. We do so by giving a realistic example. The method has also been successfully applied to dozens of meshes on complex geometries. Published in 2003 by John Wiley & Sons, Ltd.

KEY WORDS: mesh smoothing; mesh optimization; hexahedral meshing; condition number; mesh quality

1. INTRODUCTION

Algorithms such as sweeping [1], whisker weaving [2], grafting [3], H-Morph [4], and hex-tet plastering [5] have been devised to create unstructured hexahedral and hex-dominant meshes. Unfortunately, these algorithms provide no quality guarantees for the meshes created. In practice, elements which are skewed and/or inverted are not uncommon with these algorithms, especially when complicated geometry is involved. Mesh quality and smoothness can be critical for both the accuracy and efficiency of simulations using various discretization techniques. Thus there is presently a need for automatic hexahedral mesh quality improvement techniques

*Correspondence to: Patrick M. Knupp, Parallel Computing Sciences Department, Sandia National Laboratories, M/S 0847, P.O. Box 5800, Albuquerque, NM 87185-0847, U.S.A.
†E-mail: pknupp@sandia.gov
‡This article is a U.S. Government work and is in the public domain in the U.S.A.

Contract/grant sponsor: Department of Energy; contract/grant number: DE-ACO4-94AL85000

Received 10 January 2002
Accepted 4 November 2002

involving node-movement, (including smoothing and optimization), node-insertion, swapping, and cleanup strategies.

Many papers have been devoted to the topic of unstructured mesh smoothing and optimization (see, for example, References [6–11]). However, the author is aware of only two other papers which specifically address unstructured *hexahedral* mesh smoothing [12, 13]. In Reference [12] the 3D elliptic grid generation equations of Winslow are discretized via a finite element method and a smoother derived therefrom. This method was implemented in the ALEGRA code for MHD applications; unfortunately, it was found that in practice, the smoother could create inverted elements. The method described in Reference [13] is based on maximizing a variant of the scaled-Jacobian metric. Neither of these approaches provide guarantees that the 'improved' mesh will (a) consist only of untangled elements and (b) actually have improved shape-quality according to the users definition. Both goals are achieved in this paper.

For finite element meshing three geometric qualities of elements are almost always important: *invertibility, size*, and *shape* . An element is invertible if it has positive local volume. If a hexahedral mesh contains inverted elements the hexahedral mesh untangling algorithm [14] is recommended to automatically remove the inverted elements by node repositioning. Assuming a mesh contains no inverted elements, element 'size' becomes the next most important metric. Element size must be small enough that discretization error is small, yet large enough that computer memory is not exceeded and the application can be solved in a reasonable amount of time.

The next important item is element 'shape'. Previous work shows that shape is a combination of element edge-length (aspect) ratios and skew [15]. Skew is a measure that gives information concerning angles within an element and is independent of edge-length ratios. Theoretical results [16] indicate that simulation accuracy is reduced if an element contains angles close to 0 or 180°. As will be shown, shape is closely connected with the element 'condition number'. A method for improving shape via the element condition number is suggested in this paper.[‡] Improvement in element condition numbers can thus be achieved by reducing element skew, improvement of aspect ratio, or both.

In Reference [17] it was shown that one way to define the shape of a tetrahedral element is in terms of the condition number of a certain matrix built from edge vectors.

A tetrahedral mesh shape-quality objective function was optimized to give well-shaped tetrahedral elements. Preliminary results for optimizing the condition number of hexahedral meshes were demonstrated in References [19, 20]. The shape of a hexahedral element can be defined in terms of the condition numbers of a set of matrices built from edge vectors [15, 21].

In the present paper the focus is exclusively on the condition number of a hexahedral element and the objective function derived from it (Sections 2 and 3). The hexahedral element shape metric guarantees that the resulting mesh remains untangled after optimization and that element shape will be improved (or at least not be degraded). Details of the optimization procedure are described in Section 4. Examples from the CUBIT meshing code are given in Section 5 reporting on performance of the method.

[‡]See References [17–19] for the first papers that described the mesh condition number.

Table I. Nodal ordering for hexahedral element.

k	(ξ,η,ζ)	x_1	x_2	x_3
0	$(0,0,0)$	1	3	4
1	$(1,0,0)$	2	0	5
2	$(1,1,0)$	3	1	6
3	$(0,1,0)$	0	2	7
4	$(0,0,1)$	7	5	0
5	$(1,0,1)$	4	6	1
6	$(1,1,1)$	5	7	2
7	$(0,1,1)$	6	4	3

2. THE SHAPE METRIC AND THE ELEMENT CONDITION NUMBER

In this section, matrices are defined which contain information about element shape, size and orientation. The matrices are used to give an abstract definition of an algebraic shape metric involving the condition number for non-simplicial elements.

We begin by subdividing a hexahedral element into the eight tetrahedra associated with the eight nodes of the hexadral element and whose edges coincide with edges of the hexahedral element. Given a hexahedral element having eight vertices ($k=0,1,\dots,7$) ordered such that the base contains nodes 0, 1, 2, and 3 and the top contains nodes 4, 5, 6, 7 (see Table I). Let x be the vector from the origin to any one of the eight vertices. Let x_1, x_2, and x_3 be the co-ordinates of the three neighbouring vertices, chosen to give proper orientation (see Table I for the proper node ordering). Form three edge vectors $e_1=x_1-x$, $e_2=x_2-x$, $e_3=x_3-x$ and define the matrix A formed from the three column vectors, $A=[e_1|e_2|e_3]$, i.e. if the components of the vector x are (x,y,z), etc., then

$$A=\begin{pmatrix} x_1-x & x_2-x & x_3-x \\ y_1-y & y_2-y & y_3-y \\ z_1-z & z_2-z & z_3-z \end{pmatrix}$$

There are eight such matrices, A_k—one for each vertex of the hexahedron. It is assumed that the element is untangled; i.e. $\alpha_k=\det(A_k)>0$ for all k.[§] If any element of the mesh is tangled, then one must untangle the mesh before optimizing shape using, for example, the method described in Reference [14].[¶]

[§]To be precise, we distinguish between inverted and tangled elements. The former are elements for which the map from the master finite element to the physical element is invertible. The latter are elements for which at least one corner tetrahedron has negative volume. A tangled element is always inverted, but an inverted element is not always tangled. However, for a majority of hexahedral elements an inverted element is also tangled. It is much easier to devise a quality improvement algorithm based on the idea of tangled elements than it is to base on inverted elements since the former makes use of a finite number of tests while the latter requires an infinite number.

[¶]Untangling is automatically attempted as a pre-processing step in CUBIT when one invokes shape optimization.

In our general theory of algebraic metrics [15], there is also a set of weight matrices W_k which can be used to specify the ideal element shape. From A_k and W_k, we form matrices $T_k = A_k W_k^{-1}$ and use T_k to define 'shape' metrics. The shape metric is designed such that when the objective function constructed from the shape metrics is optimized, the matrices T_k are driven towards an orthogonal matrix. If T_k is orthogonal, then $A_k = T_k W_k$, and the element assumes the shape of the ideal element (with possibly different orientation).

The weight matrix is useful in two situations. First, the weight can vary according to element type to reflect different ideal shapes. For example, the ideal shape of a hexahedron is a cube; thus the weight matrix would be the identity matrix. For a tetrahedron, the ideal shape is an equilateral tetrahedron: this requires the weight given in Reference [17]. Second, the weight matrix is potentially useful in R-type adaptivity. In that case, one often does not know what the ideal shape should be to give the 'best' mesh for analysis purposes. In principle, the weight matrix could be adjusted from one element to another to improve simulation accuracy; we intend in the future to explore this possibility. These remarks apply to our general theory of algebraic metrics. In the present paper we are concerned with mesh improvement techniques outside the context of adaptivity. For this case, it is appropriate to select the identity matrix for the weight belonging to every element in the mesh since we are dealing with hexahedral elements.

The definition of a shape metric given in Reference [15] is repeated below. The definition reflects the fact that the shape quality metric should not depend on the size or orientation of the element, but should depend on the K matrices derived from the K vertices of the element. For convenience, the metric is scaled to have a range between zero and one. The metric equals one if and only if the matrices T_k are scalar multiples of orthogonal matrices. In that case, the element shape is a scaled rotation of the ideal element shape. Finally, the metric is zero if and only if the three edges at one vertex are coplanar. Shape is not defined for tangled elements.

Definition
Let f be an algebraic mesh quality metric.[‖] Then f is a non-simplicial algebraic *shape* metric if

- the domain of f is the set of matrices $T_k = A_k W_k^{-1}$, $k = 0, 1, \ldots, K-1$, with $\det(T_k) \geqslant 0$,
- f is scale and orientation invariant,
- $0 \leqslant f(\{T_k\}) \leqslant 1$, for all T_k,
- $f(\{T_k\}) = 1$ if and only if T_k is a scalar multiple of an orthogonal matrix for all k,
- $f(\{T_k\}) = 0$ if and only if $\det(T_k) = 0$ for some k.

The notation $f(\{T_k\})$ is shorthand for f being a function of all K matrices of the type T_k. Let the condition number be $\kappa(T) = |T|\,|T^{-1}|$.[**]

Proposition
$f = 8/\sum_k(\kappa(T_k)/3)^2$ is an algebraic shape metric for hexahedral elements.

[‖]See [15] for the definition of an algebraic mesh quality metric and for technical details on the items mentioned below.

[**]As explained in our previous work, we use the Frobenius matrix norm in the definition of condition number. This is not necessary, but has the advantage of a straightforward geometrical interpretation.

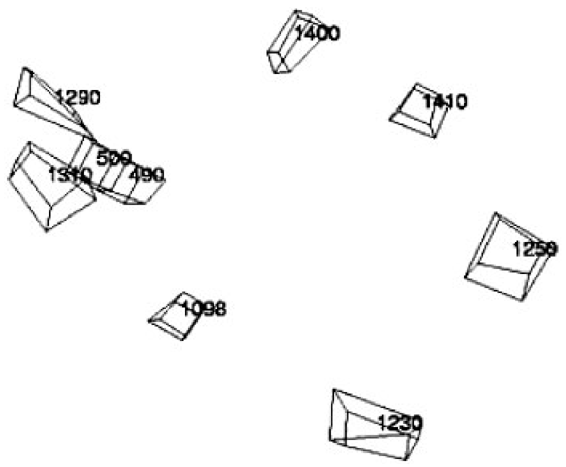


Figure 1. Hexahedral elements whose shape quality is given in Table II.

Table II. Shape quality of hexahedral elements drawn in Figure 1.

Hex ID	Shape
1290	0.15
1310	0.31
1250	0.40
1230	0.51
1400	0.61
1410	0.70
1098	0.82
490	0.90
500	0.97

Proof
The first item in the definition is immediate. For the second item, note that if ρ is a positive scalar and R is an orthogonal matrix, then $\kappa(\{\rho R T_k\}) = \kappa(\{T_k\})$, hence f is scale and orientation invariant. It has been shown in previous papers that $1 \leqslant \kappa/3 < \infty$; the third item follows directly. For the fourth item, $T_k \in SR(n)$ means that there exists a scalar ρ and an orthogonal R such that $T_k = \rho R$. If this is true for each k, then $f = 1$. On the other hand, if $f = 1$, then the maximum value of f is attained. Expressing f in terms of the singular values of each T_k, one finds the maximum is attained when all the singular values are equal (see Reference [15] for details). From this, one must have $T_k \in SR(n)$. Finally, T_k degenerate for some k means that $\tau_k = 0$. Hence $\kappa(T_k) \to \infty$ and so $f \to 0$. The reverse can be shown using singular values. $\square$

There is more than one function f that satisfies the definition of a shape metric. For example, shape can be defined by modified Winslow ($f = 3/(\tau^{2/3}|T^{-1}|^2)$), mean ratio ($f = 3/(\tau^{-2/3}|T|^2)$), and inverse condition number ($f = 3/\kappa(T)$). Any such shape metric can be used to build an objective function to improve shape quality. The present paper focuses on the condition number shape metric.

To develop geometric intuition for the hexahedral shape metric based on condition number see Figure 1 and Table II which show nine hexahedral elements of varying shape quality. The best shaped element is number 500, with a shape of 0.97, while the worst is number 1290, with a shape of 0.15. The latter element can be seen in the figure to be badly pinched off on the right end. Generally speaking, experience has shown that elements whose shape is greater than about 0.2 represent geometrically well-shaped elements.

3. THE CONDITION NUMBER OBJECTIVE FUNCTION

An objective function based on the shape quality metric described in the previous section is now constructed. The objective function measures the overall shape quality of all the elements in a given set of hexahedra.

Let $T_{n,k}$ be the matrix corresponding to the kth node of the nth hexahedral element ε_n. Define

$$f_n = f(\varepsilon_n) = \frac{1}{\frac{1}{8}\sum_k (\kappa(T_{n,k})/3)^2}$$

to be the shape quality metric of the nth hexahedral element of the mesh ($n = 1, \ldots, N$).

In constructing the objective function, it is best to work with $1/f_n$ instead of f_n because the former approaches infinity as untangled elements approach the set of tangled elements. This provides a greater numerical range and a steeper gradient than if working directly with shape metrics. The objective function is the following sum over each element ε_n in a given set ε of hexahedral mesh elements

$$\begin{aligned} F &= \frac{1}{N}\sum_n (1/f_n) - 1 \\ &= \frac{1}{8N}\sum_n \sum_k (\kappa(T_{n,k})/3)^2 - 1 \end{aligned}$$

This is basically the sum of the squares of the element condition numbers. For convenience, the objective function has been scaled so that the minimum value of F is zero. F is a function of the co-ordinates of the interior nodes of ε; positions of nodes on the boundary of ε are assumed to be fixed. Because it is assumed that the initial mesh elements are never tangled, it is proper to state that F is minimized *with the constraints* $\det(T_{n,k}) > 0$. As will be seen in the next section, it is possible to avoid the use of a true constrained minimization procedure even though this is the proper mathematical statement of the problem. By doing so, the author believes that the minimum can be found faster than if a constrained minimization were performed.

The set of mesh elements ε could be all of the hexahedral elements in the entire mesh of a volume or some subset thereof, depending on what set of elements needs improved quality. Note in particular that the domain of the objective function need not (but could) be restricted to the ball of elements attached to a particular node of the mesh as is done in so-called

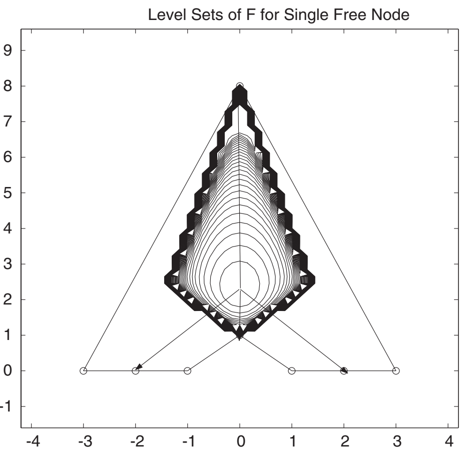


Figure 2. Level sets of F for mesh with one free node and four quadrilaterals (first boundary node configuration).

'local' optimization (see [6, 8] for examples of local optimization).[††] A formal statement of the optimization problem is thus to

Minimize F as a function of all the free node co-ordinates of the mesh, subject to the constraints that the volumes $\det(T_{n,k})$ *are positive and that the nodes on the boundary of the mesh are fixed.*

It is assumed in this approach that the initial mesh satisfies the constraint that the volumes $\det(T_{n,k})$ are positive. If they are not positive, the mesh is commonly said to be tangled. In that case we apply the mesh untangling algorithm described in Reference [14] to procure an untangled initial mesh.

To develop intuition concerning this objective function, the level sets of F for one free node attached to four *quadrilateral* elements are plotted in Figures 2 and 3.[‡‡] The level sets appear to be convex, indicating a unique minimum. The irregular contours at boundary of the feasible region are graphical artifacts stemming from the fact that the objective function is approaching infinity. The objective function was set to zero for points outside the feasible region. A dozen different boundary node configurations were tried; all appeared to give convex level sets. This suggests that, for a single free node in a quadrilateral mesh, the objective function always has convex level sets. No attempt has been made to prove that this is true for quadrilateral meshes in general. Even if it were true, we would like also to know if it were true for hexahedral meshes. It is also not known if the objective function for quadrilaterals or hexahedra is convex for more than one free node. Nevertheless, good results are obtained for hexahedral meshes with this objective function, as will be seen in Section 5.

[††]The set of elements need not even form a connected set.

[‡‡]For quadrilateral elements,

$$F = \frac{1}{4N} \sum_n \sum_k (\kappa(T_{n,k})/2)^2 - 1$$

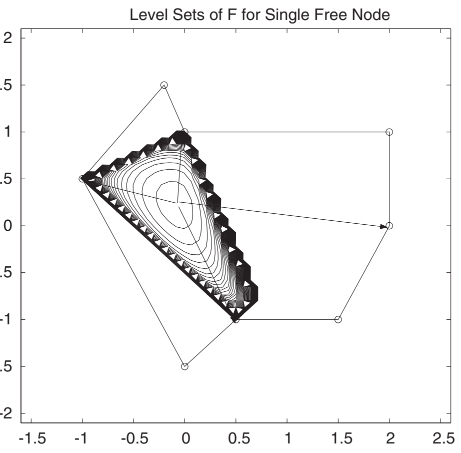


Figure 3. Level sets of F for mesh with one free node and four quadrilaterals (second boundary node configuration).

The objective function can be expressed as an appropriate scaling of the ℓ_2 norm of the vector of $\kappa(T_{n,k})/3$'s. Thus, optimization will improve the average element quality and not necessarily the worst case quality. Techniques for emphasizing the worst quality elements are discussed in the next section.

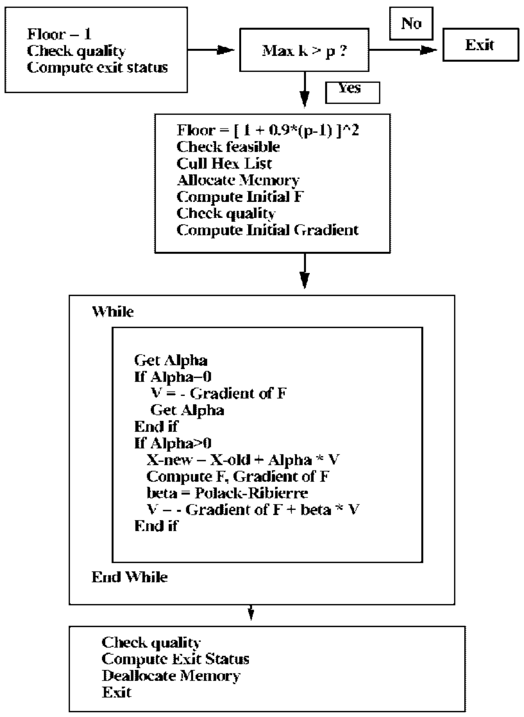
4. OPTIMIZATION PROCEDURE

The optimization procedure used to minimize the objective function consists of a conjugate gradient and line-search method based on the presentation in Reference [22]. The procedure differs from an ordinary conjugate gradient method in a number of respects which will become clear in the description below. The goal of the procedure is to find a local minimum at which the gradient is zero; no attempt is made to find the global minimum.

Space is lacking for a detailed description of the individual components of the optimization procedure, but an overview of the method is shown in Figure 4. There are two user input parameters: p and q, which act as stopping criteria.

The procedure begins by first checking the quality of the initial mesh: the maximum condition number over all the input hexahedral elements is computed. In exit status it is determined if this maximum value is less than p (default value is 2.0). The procedure is halted if the maximum condition number of the elements of the mesh is less than the user-specified criterion p. Otherwise, the initialization phase begins.

To force the procedure to emphasize the worst quality elements instead of the average, we make use of a 'floor' parameter. To initialize, first set the 'floor' parameter equal to p^2, based on the input value of p. The 'floor' represents the square of the condition number. If $\kappa^2(\varepsilon_n) < \text{floor}$, the value 'floor' is returned instead of the true condition number (squared). The effect of this is to cause the optimization procedure to not move mesh nodes adjacent to elements whose condition number is below p.



```
Floor – 1
Check quality
Compute exit status

Max k > p ?

Floor = [ 1 + 0.9*(p-1) ]^2
Check feasible
Cull Hex List
Allocate Memory
Compute Initial F
Check quality
Compute Initial Gradient

While
    Get Alpha
    If Alpha=0
      V = - Gradient of F
      Get Alpha
    End if
    If Alpha>0
      X-new = X-old + Alpha * V
      Compute F, Gradient of F
      beta = Polack-Ribierre
      V = - Gradient of F + beta * V
    End if
End While

Check quality
Compute Exit Status
Deallocate Memory
Exit
```

Figure 4. Condition number optimization flow chart 2.

The initial mesh is then verified to lie within the feasible region, i.e. that every element is untangled. This is to satisfy the basic assumption in this paper that one starts with an untangled mesh. If the assumption is not satisfied the optimal mesh may be tangled. This feasibility check is also performed in the get-alpha and gradient functions to indirectly enforce the constrained minimization. In get-alpha, if a trial location results in an tangled element, the location is rejected.

The initial list of hexahedral elements is then culled by eliminating those hexes whose condition number is already less than p. The culled hex list contains those elements whose quality exceeds p and should be improved. The list also includes all of the neighbouring elements of the hexahedra needing improvement. The reason for culling the list is to emphasize the worst quality elements and to reduce the size of the overall computation in order to be faster.

Memory is allocated for 12 one-variable arrays of length N, where N is the number of free nodes (i.e. nodes that are allowed to move during the minimization). The arrays contain node positions, new and old gradients and the search direction V.

The value of the objective function for the initial mesh is computed next, along with the initial gradient. Gradients are computed numerically. This ends the initialization phase.

Optimization is controlled through a 'while' loop which is executed as long as (1) the user-specified CPU time limit, q, is not exceeded, (2) the step-size 'alpha' is greater than zero (i.e. a true descent direction has been found), and (3) the maximum condition number exceeds p. Within the 'while' loop a line search procedure (get alpha) is performed to determine the step-length in the given search direction. The parameter alpha, which gives a step-length that significantly reduces the value of the objective function, is returned.

If alpha is zero, then one is potentially at a local minimum. To verify this is true, the search direction is changed from V (conjugate gradient) to minus the gradient of F (steepest descent). The line search is repeated for the new search direction. If alpha is still zero, then the while loop is halted because a local minimum has been found.

If alpha is greater than zero then the node positions are updated using the step-length and search direction. The value of the objective function and gradient for the new node positions is computed. The parameter beta in the conjugate gradient method is computed using the Polack–Ribiere formula [22]. The search direction V is updated using the gradient of the objective function, beta, and the old search direction. This ends the 'while' loop.

The optimization procedure ends with a final check of mesh quality, printing of exit status messages, and de-allocation of memory.

5. NUMERICAL RESULTS

We have tested this optimization algorithm on a unit cube containing a structured mesh of $10 \times 10 \times 10$ hexahedral elements. If the initial mesh is the uniform structured grid, the optimization procedure immediately halts since this is already the minimum. If the initial mesh is a random perturbation of the nodes of the uniform mesh, the optimization procedure is able to converge to the uniform mesh.

A realistic example is based on an assembly mesh created by the GraftTool [3] and sweeping. Figure 5 shows six sweepable volumes converging to a central volume shown

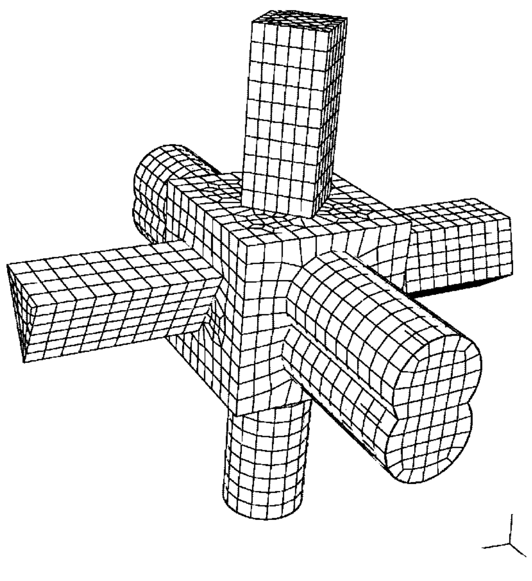


Figure 5. Multi-volume assembly mesh.

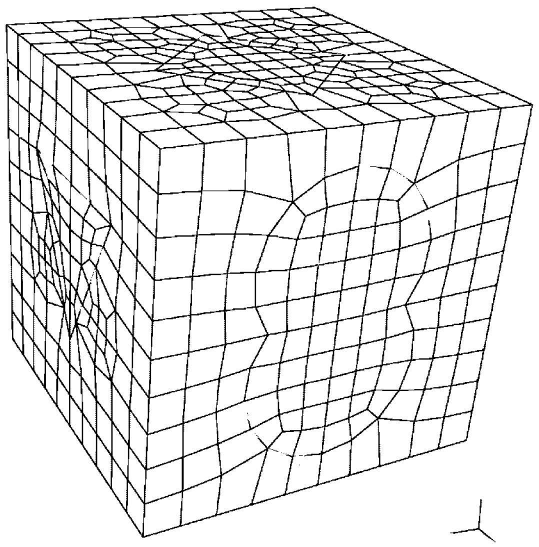


Figure 6. Surface mesh on grafted volume.

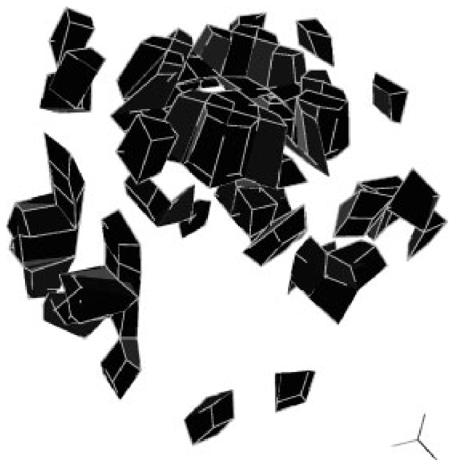


Figure 7. Hex elements in grafted volume with condition number greater than 2.4.

Table III. Optimization procedure summary.

Iteration	F	Max Grad	$R/3$	Max $\kappa/3$	Min JSC
1	4.182	30.32	1.67	6.96	0.1265
2	4.157	75.46	1.67	6.76	0.2011
3	4.155	48.72	1.66	4.49	0.1771
4	4.139	18.53	1.61	3.49	0.3144
5	4.116	6.03	1.61	2.92	0.3264
6	4.113	2.48	1.60	2.60	0.3681
7	4.110	3.28	1.60	2.74	0.3485
8	4.110	2.56	1.60	2.71	0.3474
9	4.109	2.40	1.60	2.46	0.3681
10	4.108	1.06	1.60	2.37	0.3681

in Figure 6. The example is deceptively simple since the geometry is that of a brick. However, the mesh inside has a quite complicated topological structure that enables one to transition between swept meshes having completely different connectivities. This point should become obvious upon close inspection of the surface meshes in Figure 6. Improvement of the shape quality of the mesh in the central volume, which contains 1458 hexahedral elements with a non-trivial unstructured connectivity, is sought. The average condition number of the mesh (before culling) is 1.472 while the maximum (worst-case) is 6.482. There are 88 elements whose condition number exceeds 2.4 (see Figure 7). Culling with a condition number of 2.4 reduces the number of hexahedral elements in the HexList from 1458 to 798 and the number of free nodes from 1081 to 294. Table III shows the behaviour of the quality metrics during the optimization procedure. The first column in the table is the iteration count (one

Table IV. CPU time vs mesh size (floor = 2).

Size	Total hexes	Hexes not culled	Number of nodes	CPU (s)
1	1000	995	708	22.4
0.9	1331	1318	949	37.0
0.8	2197	2186	1664	75.3
0.7	2744	2733	2142	111
0.6	4913	4899	4014	207
0.5	8000	7990	6749	395

iteration per search direction). The second column gives the value of the objective function; note that this is not reduced very much and is nowhere close to zero (all elements perfectly shaped). This is partly because the method emphasizes the worst case elements instead of the average. The third column shows the maximum absolute value of any component of the gradient; the gradient gradually decreases towards zero as a stationary point is approached but does not do so monotonically. The fourth column shows the average value of the condition number (divided by 3); this decreases very slowly and monotonically. The maximum condition number, given in column five, decreases rapidly from near 7 to the input cutoff value of 2.4; when this is attained, the iteration halts. The last column gives the value of the minimum scaled-Jacobian;[§§] this increases non-monotonically from 0.13 to 0.37. Thus, a significant improvement in the quality of the mesh was achieved in 22 CPU seconds on a SUN Ultra 2 Workstation. After optimization, there were no elements with condition number exceeding 2.4. For comparison, we applied Laplacian smoothing to this example, with a tight tolerance to ensure a converged solution. Laplacian smoothing resulted in a worst-case condition number of 4.9; this is much worse quality than the 2.2 achieved by direct condition number optimization.

To study the effect of mesh size on CPU time a brick of size 10 was meshed with sizes ranging from 1 to 0.5, resulting in 1000 to 8000 hexahedral elements. The mesh was optimized with a floor of 2. Results are shown in Table IV. The first column of the table is the average size of each element (relative to the size of the brick). The second column is the number of hexahedral elements in the brick while the third column is the number of hexahedral elements after culling. The fourth column is the number of nodes in the mesh after culling. The last column gives the CPU time in seconds that it took to improve all elements so that the worst condition number was less than 2. For small problems, the optimization method gives improved meshes in a reasonable amount of time. Meshes of up to 10 000 elements can be smoothed in under 8 min. The method has not been tried for very large meshes. A timing model for CPU time (in seconds) vs the number of nodes (N) of the form $\mathrm{CPU} = 0.01N^{1.2}$, gives a very good fit to the data. According to this model, it would take 44 h to optimize a mesh with one million nodes on our workstation. The speed of this algorithm is slow compared to standard smoothers such as Laplace smoothing. From that perspective, there is a tradeoff between robustness and speed: Laplace smoothing is fast but non-robust while condition number smoothing is robust but not fast. For this reason, plans for the future include

[§§]The minimum scaled-Jacobian (JSC) ranges from -1 to $+1$, with negative values signifying tangled elements and a perfect cube having a value of $+1$.

a study of alternate minimization techniques to determine if the minimum condition number can be found faster.

6. SUMMARY AND CONCLUSIONS

A method for improving mesh shape quality by node-movement is described which guarantees that elements will not be tangled and that the quality will be improved or at least not degraded. The method is based on minimization of an objective function constructed from a shape quality metric for hexahedral elements. The shape quality metric, in turn, is based on the condition number of a weighted Jacobian matrix formed from element edge vectors. The shape metric is scale and orientation invariant so that minimization of the objective function results in meshes with improved angles and aspect ratios while ignoring element size and orientation. Minimization requires that the initial mesh contains no tangled elements. The minimization is indirectly constrained to the set of untangled elements by rejecting node movements which would result in element tangling. The constraint is consistent with the fact that the objective function tends to infinity as elements approach a tangled configuration. The objective function can include all the elements in a given mesh (global), only those connected to a particular node (local), or some intermediate case obtained by culling hexes from the global list. A conjugate gradient method with line search, modified to handle the constraint and to emphasize the worst quality element, is used as the optimization procedure. The procedure has proven to be robust in that no tangled elements are created and quality is indeed improved. We emphasize that *although only one numerical example is given in this paper, the algorithm has been successfully used on dozens of meshes of up to 10 000 elements generated on complex geometries*, thus demonstrating that the algorithm is robust and practical for small problems. In the future, methods to improve the speed of the optimization procedure will be explored. Possible performance gains may come from improvements in the line search or from alternative minimization procedures such as trust region methods.

ACKNOWLEDGEMENTS

This work was funded by the Department of Energy's Mathematics, information and Computational Sciences Program (SC-31) and was performed at Sandia National Laboratories. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-ACO4-94AL85000.

REFERENCES

1. Knupp P. Next-generation sweep tool: a method for generating all-hex meshes on two-and-one-half dimensional geometries. *7th International Meshing Roundtable*, Detroit MI, October 1998; 505–513.
2. Folwell N, Mitchell S. Reliable whisker weaving via curve contraction. *7th International Meshing Roundtable*, Dearborn, MI, 1998; 365–378.
3. Jankovich S, Benzley S, Shepherd J, Mitchell S. The graft tool. *8th International Meshing Roundtable*, S. Lake Tahoe, CA, 1999; 387–392.
4. Owen S, Saigal S. H-Morph: an indirect approach to advancing front hex-meshing. *International Journal for Numerical Methods in Engineering* 2000; **49**(1/2):289–312.
5. Meyers R, Tautges T, Tuchinsky P. The Hex-Tet dominant meshing algorithms as implemented in CUBIT. *7th International Meshing Roundtable*, Dearborn, MI, 1998; 151–158.

6. Canann S, Tristano J, Staten M. An approach to combined laplacian and optimization-based smoothing for triangular, quadrilateral, and quad-dominant meshes. *7th International Meshing RoundTable*, Dearborn, MI, 26–28 October 1998; 479–494.
7. Chen CL, Szema KY, Chakravarthy SR. Optimization of unstructured grids. AIAA 95-0217, *33rd Aerospace Sciences Meeting and Exhibit*. Reno, NV, 9–12 January 1995.
8. Freitag L, Jones M, Plassmann P. An efficient parallel algorithm for mesh smoothing. *4th International Meshing RoundTable*, Albuquerque, 1995; 47–58.
9. Hansbo P. Generalized laplacian smoothing of unstructured grids. *Communications in Numerical Methods in Engineering* 1995; **11**:455–464.
10. Parthasarathy N, Kodiyalam S. A constrained optimization approach to finite element mesh smoothing. *Finite Elements in Analysis and Design* 1991; **9**:309–320.
11. Zavattier P. Optimization strategies in unstructured mesh generation. *International Journal for Numerical Methods in Engineering* 1996; **39**:2055–2071.
12. Tipton R. Grid optimization by equipotential relaxation. Unpublished manuscript, Lawrence Livermore National Laboratory, 28 November 1990.
13. Calvo N, Idelsohn S. All-hexahedral mesh smoothing with a node-based measure of quality. *International Journal for Numerical Methods in Engineering* 2001; **50**(8):1957–1967.
14. Knupp P. Hexahedral and tetrahedral mesh untangling. *Engineering with Computers*, 2001; **17**(3):261–268.
15. Knupp P. Algebraic mesh quality measures. *SIAM Journal on Scientific Computing* 2001; **23**(1):193–218.
16. Babuska I, Aziz A. On the angle condition in the finite element method. *SIAM Journal on Numerical Analysis* 1976; **13**:214–226.
17. Freitag L, Knupp P. Tetrahedral element shape optimization via the Jacobian determinant and condition number. *Proceedings of the 8th International Meshing RoundTable*, Lake Tahoe, CA, 1999; 247–258.
18. Knupp P. Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities, Part II—a frame-work for volume mesh optimization. *International Journal for Numerical Methods in Engineering* 2000; **48**:1165–1185.
19. Knupp P. Matrix norms and the condition number. *Proceedings of the 8th International Meshing Roundtable*, Lake Tahoe, CA, 1999; 13–22.
20. Kober C, Muller-Hannemann M. Hexahedral mesh generation for the simulation of the human mandible. *Proceedings of the 9th International Meshing Roundtable*, New Orleans, LA, 2000; 423–434.
21. Knupp P. Algebraic mesh quality metrics for unstructured initial meshes. *Finite Elements in Analysis and Design* 2003; **39**:217–241.
22. Nocedal J, Wright S. *Numerical Optimization*. Springer: New York, 1999.