JOURNAL OF THEORETICAL
AND APPLIED MECHANICS
52, 1, pp. 151-163, Warsaw 2014

A COMPARISON OF OBJECTIVE FUNCTIONS OF OPTIMIZATION-BASED SMOOTHING ALGORITHM FOR TETRAHEDRAL MESH IMPROVEMENT

Cui Dai, Hou-Lin Liu, Liang Dong

Jiangsu University, Research Center of Fluid Machinery Engineering and Technology, Jiangsu, China

e-mail: liuhoulin@ujs.edu.cn

The objective function based on mesh quality metric has a major impact on smoothing unstructured tetrahedral meshes. The ability of seven mesh quality metrics to distinguish four kinds of poor-quality elements and their effects on the change of element shape are analyzed in detail. Then, four better mesh quality metrics are chosen to construct objective functions. In addition, the rational determination of searching direction and the optimal step size in the optimization algorithm of solving the objective function are proposed. Finally, comparisons with the other three objective functions are made according to different number of elements, iteration limit, and the desired accuracy in the improved mesh. It is found that with the increase of the number of elements, the time consumed during optimization increases, but the changes of the worst quality element are different. The number of iterations has little effect on the mesh quality and the time cost. The increasing of the desired degree of accuracy will improve the mesh quality and cost more time. Furthermore, the approach using objective function is compared with Freitag's common approach. It is clearly shown that it performs better than the existing approach.

Key words: objective function, mesh quality metric, optimization-based smoothing, mesh quality improvement, mesh generation

1. Introduction

The finite element and finite volume methods are invaluable tools for solving complex engineering problems in structural analysis, fluid dynamics, electromagnetism, and many other areas (Dobrzynski and Frey, 2008; Montenegro *et al.*, 2009). These tools rely on the mesh, a discretization of space into simple geometric pieces that makes numerical solution possible. Tetrahedral meshes are a popular choice for discretization of three-dimensional domains, and can be generated by advancing front techniques, Delaunay or Octtree methods. However, not every mesh is suitable for numerical computation. Poorly-shaped tetrahedra in a mesh can result in numerical errors and increase the time cost to find the solution (Munson, 2007; Park and Shontz, 2010). Hence, there is a market for mesh improvement tools which can make the existing tetrahedra conform to a given domain together with certain constraints for the size and shape of the elements.

Mesh improvement techniques can roughly be classified into two types of methods that modify mesh topology and those which do not. The first modifies topology by inserting or deleting nodes as well as changing connectivity of nodes (Edelsbrunner and Shah, 1996). In contrast, the second, known as the smoothing method (Xu *et al.*, 2009; Tournois *et al.*, 2009), preserves mesh topology by applying appropriate node placement techniques. What we concern in this context are node repositioning algorithms that preserve mesh topology. One of the most common smoothing techniques is Laplacian smoothing, which relocates a single point to geometric center of its directly connected neighboring nodes. This technique is computationally inexpensive and simple to implement. But, it is less efficient in the case of tetrahedral meshes, since the variety of adverse topological and geometrical configurations increases in 3D, which makes the Laplacian smoothing fail (Mao *et al.*, 2006). To address these problems, researchers have developed

optimization-based methods (Hetmaniuk and Knupp, 2011; Vartziotis *et al.*, 2009; Dong *et al.*, 2011) geared towards improving the element quality. The methods are formulated in terms of design variables (one or more nodes to be repositioned), an improvement goal (quality metric, objective function, and constraints), and the algorithm used to calculate an optimal solution. In them, an objective function based on a suitable quality metric has a crucial impact on the solution accuracy and efficiency. Unfortunately, little is known about the relative merits of using one objective function over another in order to smooth a particular unstructured mesh. For example, it is not known in advance which objective function will converge to an optimal mesh faster or which objective function will yield a mesh with better quality in a given amount of time. One objective function may converge faster than others, whereas another objective function may improve the quality of meshes with heterogeneous elements more quickly than its competitors.

To answer the above questions, a study that compares seven quality metrics is conducted to clarify their abilities of identifying poor-quality elements and assessing the change of element shape. Then, the objective functions are investigated to represent the overall mesh quality measured by quality metrics. After that, Section 3 gives our improvement to the optimization algorithm. In Section 4, the factors that affect the optimizing effect, such as the element number, iteration limit and the desired accuracy in the resulting mesh, are discussed. And some experimental results of our method and comparisons with other methods are given. In the last Section, we will conclude our discussion.

2. Tetrahedral mesh quality metrics

The direct measure of mesh quality is to see the precision and speed of numerical solution using the mesh. Obviously, it cannot directly be used for the examination and improvement of mesh the quality. Therefore, different quality metrics were raised by researchers from various respects.

In many mesh quality metrics, their properties have been assumed or stated without proof. Although those metrics claim that their properties appear obvious, we believe they should be verified rigorously. We consider several commonly used tetrahedral mesh quality metrics as shown in Table 1. The range of the quality metrics is in the interval $[0, 1]$. Each metric attains a maximum value only for the regular tetrahedron. Larger values of the metrics represent good quality tetrahedra (close to a regular tetrahedron) and smaller values represent poor-quality tetrahedra (close to degenerate). Our goal is to provide a number of useful results on tetrahedral mesh quality metrics that may lead to a better assessment of tetrahedron, and get better objective functions which are deduced by those metrics for mesh quality improvement.

Table 1. Different quality metrics

Label	Expression	Range	Used in reference
q_1	$6\sqrt{6}V/\Big[\Big(\sum\limits_{i=1}^{4} S_i\Big) \max\limits_{j=1,\ldots,6} L_j\Big]$	$[0,1]$	Geuzaine and Remacle (2009)
q_2	$V_{max} - V)/V_{max}$	$[0,1]$	Lo (1997)
q_3	$6\sqrt{2}V/\left(\sqrt{\frac{1}{6}\sum\limits_{j=1}^{6} L_j^2}\right)^3$	$[0,1]$	Nie (2003)
q_4	$3r/R$	$[0,1]$	Parthasarathy *et al.* (1994)
q_5	L_{min}/L_{max}	$[0,1]$	Shewchuk (2002)
q_6	$\min(\theta_1, \theta_2, \theta_3, \theta_4)$	$[0,1]$	Si and Gärtner (2011)
q_7	$1 - \max\Big(\frac{Q_{max}-60}{120}, \frac{60-Q_{min}}{60}\Big)$	$[0,1]$	Lo (1997)

Nomenclature: V is the tetrahedral volume, V_{max} is the maximum volume of an equilateral cell whose circumscribing radius is identical to that of the mesh element. L_j, $j = 1, \ldots, 6$ are its

edge lengths, L_{min} is $\min L_j$, L_{max} is $\max L_j$. S_i is the surface area of a triangular facet, r is insphere radius, R is circumsphere radius, Q_{max} and Q_{min} are the maximum and minimum angles between the edges of the element, l_{ij} is the length of the edge joining vertices i and

$$\theta = 2\arcsin\frac{12V}{\sqrt{\prod\limits_{\substack{j,k\neq i\\ 1\leqslant j\leqslant k\leqslant 4}}[(l_{ij}+l_{ik})^2 - l_{jk}^2]}}$$

2.1. Numerical tests

To find out whether quality metrics are equivalent, mainly two aspects are considered. The first is the number of nodes to be moved, and the second is the evaluation made on four kinds of poor-quality tetrahedra (see Fig. 1). Furthermore, mesh improvement approaches normally move one node at each iteration. However in the paper, the conditions of moving one node, two and three nodes are all investigated.

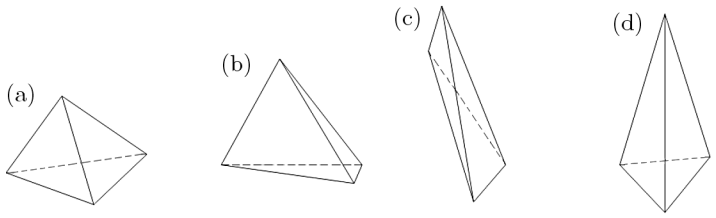

Fig. 1. Some poor-quality tetrahedra: (a) no short edges, but four nodes are nearly coplanar, (b) only one short edge, (c) only two relatively short edges, (d) three shorter edges are in the same plane

Specific test cases are designed, as shown in Table 2. Let (t_0, t_1, t_2, t_3) denote a tetrahedron with four pairwise disjoint nodes $t_i \in R^3$, $i \in \{0, \ldots, 3\}$, which is positively oriented (as shown in Fig. 2). Let u control each node's position in the tetrahedron (see in Table 2), where $0 < u \leqslant 1$. When $u = 1$, it is a regular tetrahedron.

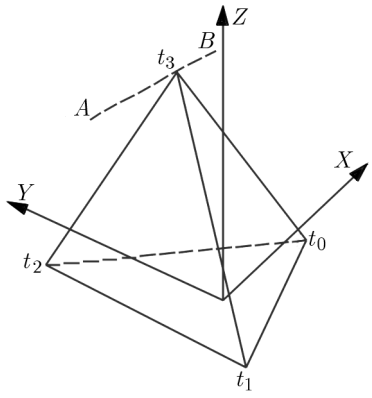

Fig. 2. Sketch of a tetrahedral element

2.2. Numerical results

Results of this numerical test with respect to node movement are depicted in Figs. 3 and 4. It can be indicated that mesh quality vary significantly for a given quality metric with movement of the nodes. It means that the node position and hence the geometry of the tetrahedron has big influence on the quality metric. For q_5 and q_7 in Fig. 3, when u approaches to zero, their values grow larger ($q_5 = 0.578792$, $q_7 = 0.504104$), so they cannot judge the tetrahedron whose four nodes are nearly coplanar. Such type of tetrahedrons may be considered as a well-shaped

Table 2. Test cases

No.	Coordinates for each node of tetrahedron	No. of nodes moved	Instruction
1	$t_0(0,0,0)$, $t_1(1,0,0)$, $t_2(0.5,\sqrt{3}/2,0)$, $t_3(0.5,\sqrt{3}/6,\sqrt{6}u/3)$	1	when $u \to 0$, 4 nodes are nearly coplanar, see Fig. 1a
2	$t_0(0,0,0)$, $t_1(1,0,0)$, $t_2(0.5,\sqrt{3}u/2,0)$, $t_3(0.5u,\sqrt{3}u/6,\sqrt{6}u/3)$	2	there is one short edge when $u \to 0$; nodes t_2 and t_3 are moved, at least one poor-quality triangle in the tetrahedron, see Fig. 1b
3	$t_0(0,0,0)$, $t_1(1,0,0)$, $t_2(1-0.5u,\sqrt{3}u/2,0)$, $t_3(0.5u,\sqrt{3}u/6,\sqrt{6}u/3)$	2	there are two short edges when $u \to 0$ and the nodes t_2 and t_3 are moved, at least two poor-quality triangles in the tetrahedron, see Fig. 1c
4	$t_0(0,0,0)$, $t_1(u,0,0)$, $t_2(0.5u,\sqrt{3}u/2+u/2,0)$, $t_3(0.5,-\sqrt{3}/2+2\sqrt{3}u/3,\sqrt{6}u/3)$	3	there are three short edges in the same plane when $u \to 0$ and the nodes t_1, t_2 and t_3 are moved, three poor-quality triangles in the tetrahedron, see Fig. 1d
5	$t_0(0,0,0)$, $t_1(3,0,0)$, $t_2(5,3,0)$, $A(0.6,1.2,1.0)$, $B(1.2,2.4,1.0)$	1	node t_3 moves along the line AB
6	$t_0(\sqrt{3}/3,0,0)$, $t_1(-\sqrt{3}/6,0.5,0)$, $t_2(-\sqrt{3}/6,-0.5,0)$, $t_3(0.5\cos(2\pi u),0.5\sin(2\pi u),\sqrt{6}/3)$	1	node t_3 moves in the $z=\sqrt{6}/3$ plane along the circle whose radius is 0.5

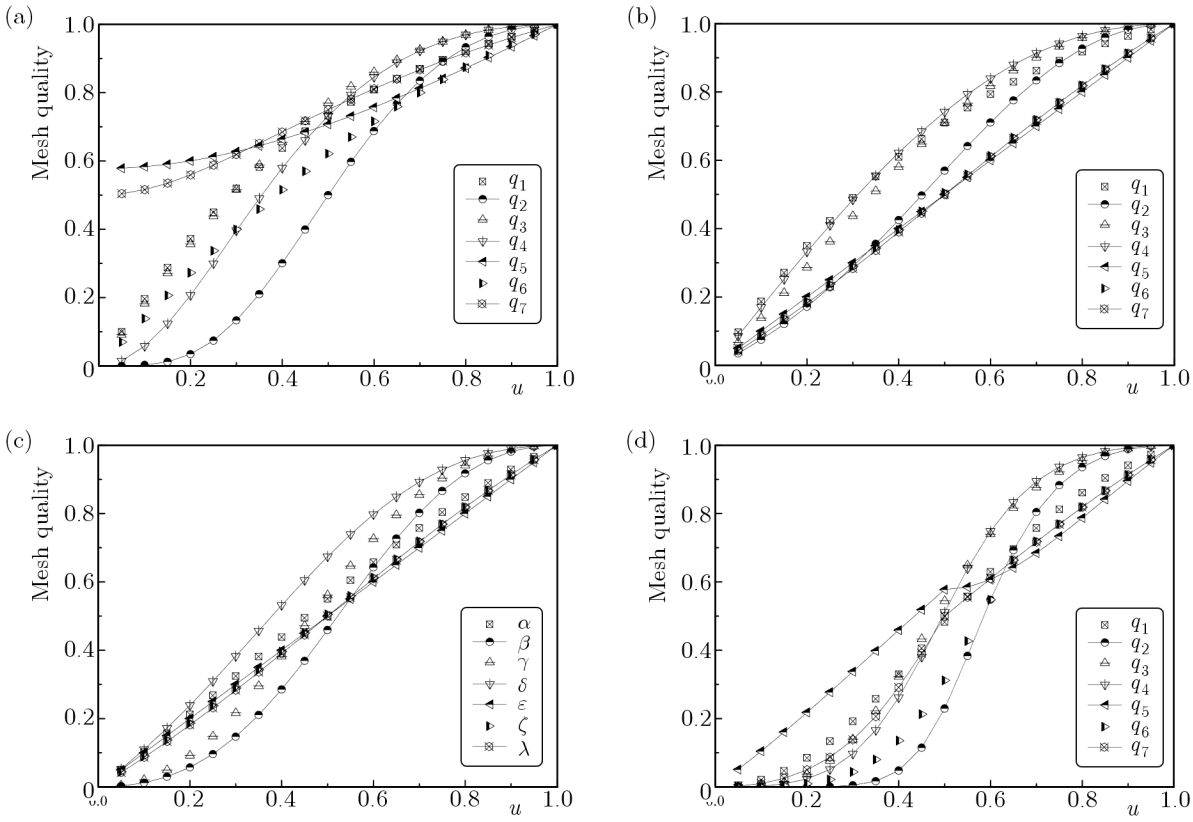

Fig. 3. Change of different metrics with u: (a) in case 1, (b) in case 2, (c) in case 3, (d) in case 4

tetrahedron during the mesh optimization, which greatly affects the quality of the final mesh. As shown in Fig. 4a, when the distance between node t_3 and A increases, the values of q_1, q_3, q_5, q_6 and q_7 increase progressively, while the values of q_2 and q_4 decrease progressively. It suggests that different mesh quality metrics might be not equivalent. In other words, the quality of elements may be good according to one metric. In contrast, it may be bad when measured by other metrics. That is, opposite optimization directions may be got by different metrics. From Fig. 4b, the values of q_1, q_5, q_6 and q_7 change periodically, while the values of q_2, q_3 and q_4 keep constant. In fact, the volume of the tetrahedron V and the length of each edge $\sum_{i=1}^{6} L_i^2$ are constant in case 6 $V = \sqrt{2}/12$, $\sum_{i=1}^{6} L_i^2 = 6.75$). So the change of u cannot result in the change of q_2, q_3 and q_4 in the case. So it can be concluded that their evaluations on the change of element shape are different, too. The metrics of q_1, q_5, q_6 and q_7 can reflect the change of element quality induced by motion of the nodes, while q_2, q_3 and q_4 cannot.

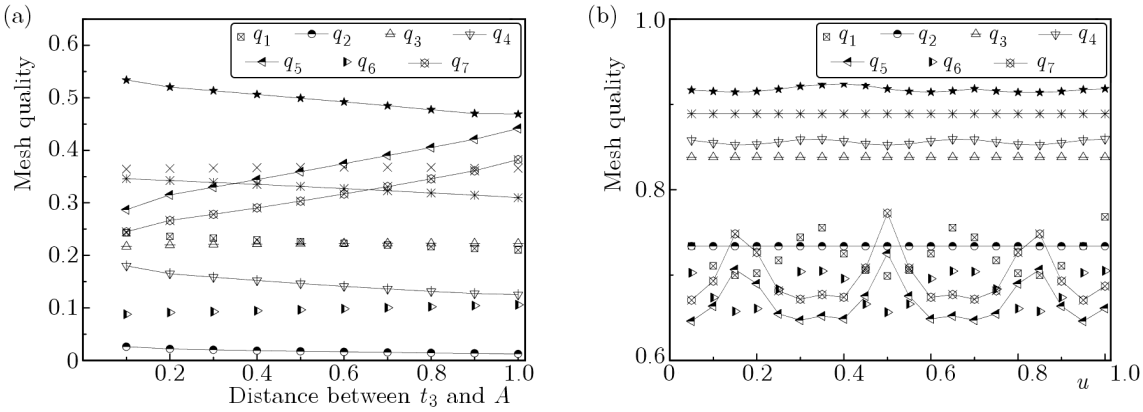

Fig. 4. Change of different metrics: (a) with the distance of t_3 and A in case 5, (b) with u in case 6

From the above analysis, we can see that q_1 and q_6 have better performances on the evaluation of poor-quality elements and the change of element shape.

2.3. Objective functions

To assess the overall quality of the mesh, both the average and the worst element quality can be adopted. However, a single bad tetrahedron can ruin a simulation: one large dihedral angle can induce an arbitrarily large and incorrect strain in the simulation of a mechanical system (Klingner and Shewchuk, 2008), hence we focus on the minimum local element quality.

We associate with the mesh a continuous function to measure the mesh quality as measured by one or more geometric properties of elements as a function of their node positions. The objective function we used is

$$f(X) = 1 - \min_{1 \leqslant i \leqslant n} q(X_i) \tag{2.1}$$

We improve the worst element quality by minimizing the objective function, where $f(X)$ is the overall mesh quality measured by the worst-quality element in the mesh, X is the position of the free node and n is the number of elements in the mesh. Let $q(X_i)$ measure the quality of the i-th element. A specific choice of q is an element quality metric.

We define the objective function as f_1 which is constructed by the quality metric q_1. In the same way, f_3, f_4 and f_6 correspond to q_3, q_4 and q_6. In order to investigate the smoothness of these typical objective functions, we let one node of a tetrahedron move only in one plane. In Fig. 5, four nodes of the two tetrahedra are fixed at the coordinates $A(0,0,0)$, $B(\sqrt{3}/2, 0.5, 0)$, $C(\sqrt{3}/2, -0.5, 0)$, and $D(\sqrt{3}/3, 0, \sqrt{6}/3)$, and the fifth node E moves freely along X and Z axes. When node E is moved, the distribution of the four objective functions is shown in Fig. 6. The lighter the region is, the lower the object functions value is. We can see that the objective function f_6 is a nonsmooth function of node positions (see Fig. 6d), because with the node movement, the identity of the objective function (and the gradient of the function) can change abruptly. Unfortunately, our smoothing algorithm (Section 3) cannot cope with this nonsmoothness. Hence, comparisons with other three objective functions are made according to different number of elements, iteration limit, and the desired accuracy in the next section.

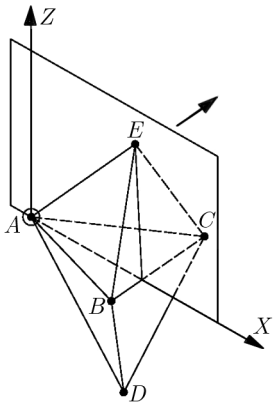

Fig. 5. Sketch of node movement in a tetrahedron

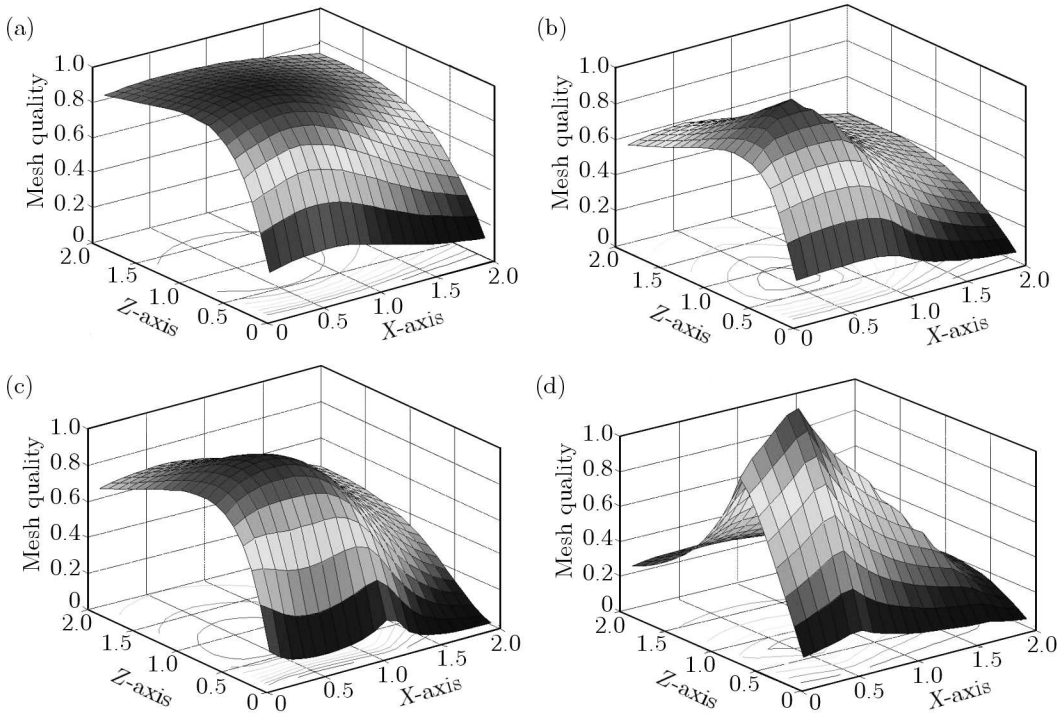

Fig. 6. Distribution for different objective functions: (a) f_1, (b) f_3, (c) f_4, (d) f_6

3. Optimization algorithm

3.1. The overall optimization algorithm

The optimization-based smoothing directly attacks the element distortion problem. Instead of applying a heuristic-based movement to each node as done in the Laplacian smoothing, the optimization-based smoothing seeks to minimize the distortion of the elements connected to each node. Let Φ denote a one-dimensional array. The overall smoothing scheme is presented. Algorithmic details for each of the major steps in this scheme are below.

Input: a given random initial mesh, objective function $f(X)$, a fixed poor-quality threshold value γ;

Output: high-quality mesh:

1) Compute the number of nodes and elements in the initial mesh
2) Calculate the quality of the initial mesh based on the chosen quality metric, and store poor-quality elements whose quality are less than γ in Φ
3) In Φ, select a poor-quality element randomly and determine its adjacent region
4) Build a locally isolated optimization region and calculate the value of the objective function in the region using Eq. (2.1)
5) Calculate the optimal solution (see Section 3.2) which is used to adjust all nodes' locations in the region simultaneously, and get the best location of X
6) Repeat steps 3 to 6 until there are no poor-quality elements in Φ.

3.2. The algorithm of calculating the optimal solution

The algorithm for calculating the optimal solution is presented in Table 3.

Table 3. Determination of optimal solution by Algorithm 1

Algorithm 1: calculate optimal solution	
1: **function** SOLUTION $(f(X), \lambda_0, \varepsilon, X_n^0, N)$	◀ $f(X)$ = objective function λ_0 = initial step size ε = desired degree of accuracy N = iteration limit X_n^0 = initial coordinate for node X
2: get free node H_n^0 and $k = 0$	◀ H_n^0 = Hessian matrix k = current iteration
3: min $f(X_n^k + \lambda_k d_n^k)$	◀ d_n^k = optimal search direction
4: compute λ_k^* using line search algorithm (see algorithm 2 in Section 3.3)	◀ λ_k^* = optimal step size
5: $X_n^{k+1} = X_n^k + \lambda_k^* d_n^k$	
6: if $\lvert f(X_n^{k+1}) - f(X_n^k)\rvert < \varepsilon$ or $k > N$ then	
7: $X_n^* \leftarrow X_n^{k+1}$ and **break**	◀ X_n^* = optimal point coordinate
8: else $\Delta X_n^{k+1} = X_n^{k+1} - X_n^k \quad \Delta g_n^k = g_n^{k+1} - g_n^k$ $H_n^{k+1} = H_n^k + \frac{\Delta X_n^k (\Delta X_n^k)^T}{(\Delta X_n^k)^T \Delta g_n^k} + \frac{H_n^k \Delta g_n^k (\Delta g_n^k)^T H_n^K}{(\Delta g_n^k)^T H_n^k \Delta g_n^k}$ $d_n^{k+1} = -H_n^{k+1} g_n^{k+1} \quad k = k + 1$ go to step 3	◀ $g_n^{k+1} = f(x)$ gradient at node X_n^{k+1}
9: **end function**	

3.3. Line search algorithm

It is well known that the line search methods play a very important role in optimization problems. We prepare for locating a local minimum in the optimization problem with no constrains. All methods have in common the basic structure. At each iteration, a direction d_n^k is chosen from the current location X_n. The next location, X_{n+1}, is the minimum of the function along the line that passes through X_n in the direction d_n^k. The line search algorithm is shown in Table 4.

Table 4. Line search algorithm

Algorithm 2: line search algorithm	
1: **function** LINE_SEARCH $\left(\varepsilon_1, \varphi(f(X), X_n^0, h_0, \alpha)\right)$	◀ $f(X)$ = objective function ε_1 = allowable error $\varphi(f(X), X_n^0, h_0, \alpha)$ = function for search region $[a_1, b_1]$ see Algorithm 3
2: compute $\mu_1 = a_1 + 0.382(b_1 - a_1)$ $\nu_1 = a_1 + 0.618(b_1 - a_1)$ and Set $i = 1$	◀ μ_1, ν_1 = initial tentative point i = current iteration
3: if $\lvert\mu_i - \nu_i\rvert < \varepsilon_1$ then	
4: return $\lambda_i^* = \frac{\mu_i + \nu_i}{2}$ and **break**	◀ λ_i^* = optimal step size
5: else if $f(\mu_i) < f(\nu_i)$ go to step 7	
6: else $f(\mu_i) \geqslant f(\nu_i)$ go to step 8	
7: set $a_{i+1} = a_i$, $b_{i+1} = \nu_i$, $\nu_{i+1} = \mu_i$, $f(\nu_i) = f(\mu_i)$	
8: compute $\mu_{i+1} = 0.618a_{i+1} + 0.382b_{i+1}$ and $f(\mu_{i+1})$	
9: $i = i + 1$ go to step 3	

In order to find the minimum of the function $f(X) : R \to R$, we need to bracket it. To bracket a minimum means fining a triple $a, b, c \in R$, $a < b < c$, such that $f(a) < f(b)$ and $f(b) < f(c)$. This indicates that the minimum is in the interval $[a, c]$. The interval search algorithm is given in Table 5.

4. Numerical experiments

The preferred objective functions may differ depending on the circumstances. The factors that may be significant in determining the subproblems are quality metric, element number, iteration limit, desired accuracy in the resulting mesh, mesh type (structured vs. unstructured), and dimension (planar vs. volume). To make the investigation manageable, we limit the number of free parameters to a fixed mesh type (unstructured), quality metric (see q_1), and optimization algorithm (see Section 3). The free parameters are the number of elements, iteration limit and desired accuracy in the resulting mesh. For each parameter to be investigated, we create a set of meshes in which we isolate the interesting parameter, allowing it to vary, while simultaneously holding the other parameters as constant as possible. A series of meshes were generated for the impeller geometry shown in Fig. 7. The machine employed for this study is equipped with Intel P4 processor (2.67 GHz). The 32-bit machine has 1 GB of RAM.

4.1. Effect of the element number

With regards to the effect of the element number on the optimization results, a series of initial meshes for the impeller with different number of nodes (V), elements (E), and the

Table 5. Search interval algorithm

Algorithm 3: search interval algorithm	
1: **function** SEARCH_INTERVAL $\left(f(X), X_n^0, h_0, \alpha\right)$	◀ $f(X)$ = objective function X_n^0 = initial coordinate for node X h_0 = initial step size α = coefficient larger than 1
2: compute $f_0 = f(X_n^0)$ and Set $j = 0$	◀ j = current iteration
3: $X_n^{j+1} = X_n^j + h_j$ and compute $f(X_n^{j+1})$	
4: if $f(X_n^{j+1}) < f(X_n^j)$ then go to step 6	
5: else go to step 7	
6: $h_{j+1} = \alpha h_j$, $X_n^* = X_n^j$, $X_n^j = X_n^{j+1}$ $f(X_n^j) = f(X_n^{j+1})$, $j = j + 1$	
7: if $j = 0$ then $h_j = -h_j$, $X_n^* = X_n^j$ go to step 3	
8: else $a = \min\{X_n^*, X_n^{j+1}\}$, $b = \max\{X_n^*, X_n^{j+1}\}$	
9: return a and b	

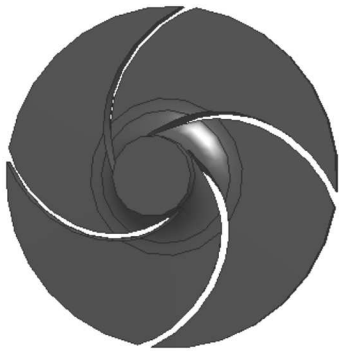

Fig. 7. Impeller model

worst quality element (M_{in}) based on q_1 were investigated, as shown in Table 6. The number of elements increases from approximately 45 000 to 400 000 (iteration limit $N = 50$, desired accuracy $\varepsilon = 10^{-5}$).

Table 6. Initial meshes for the impeller

Model	V	E	M_{in}
Impeller	9 563	45 513	0.0014
	22 680	122 027	0.0013
	38 693	214 167	0.0024
	70 128	401 696	0.0019

The final mesh quality measured by q_1 and time for three objective functions with different element numbers are acquired and plotted in Fig. 8. It is found that with the increase of the element number, the time during optimization increases for the three objective functions, while the worst element quality varies only slightly. And no matter which objective function is used in the algorithm, the mesh quality is greatly improved compared with that before the optimization. The first objective function can obtain the highest mesh quality, and the corresponding time consumed is shortest. The objective function f_3 obtains the poorest results.

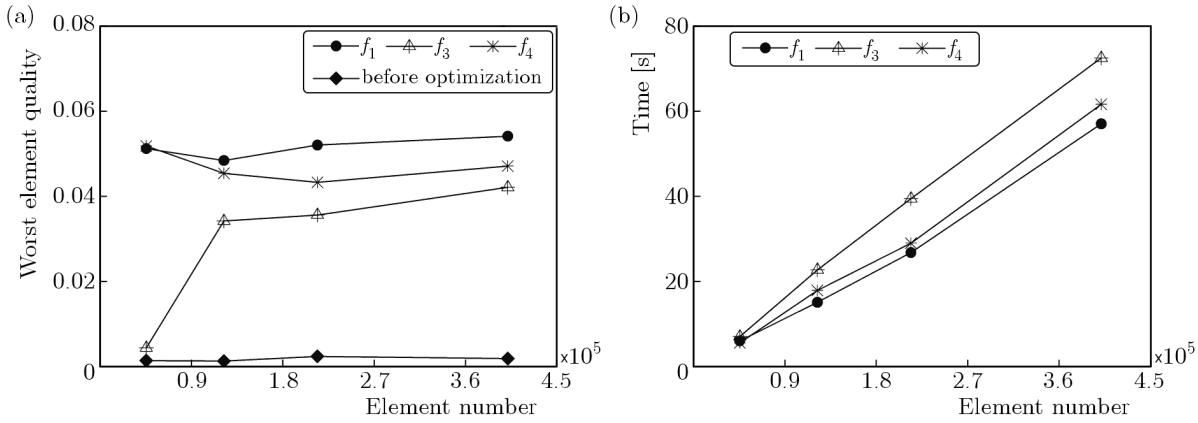

Fig. 8. Final mesh quality and time cost for different element numbers

4.2. Effect of the iteration limit

To investigate the effect of the iteration limit N on the optimization results, the third set of mesh in Table 6 is chosen, in which the number of nodes and elements are 38 693 and 214 167, respectively. In addition, the worst element quality is 0.0024. For the given random initial tetrahedron, the solution has been iteratively applied until a fixed desired accuracy of 10^{-5} has been achieved.

Figure 9 shows the mesh quality and time cost for the three objective functions with different iteration limits. It is obvious that with the increment of the iteration limit, the change of mesh quality and time are not obvious. The time consumed increases slightly only for f_3. So, the iteration limit has little effect on the mesh quality and time cost for different functions.

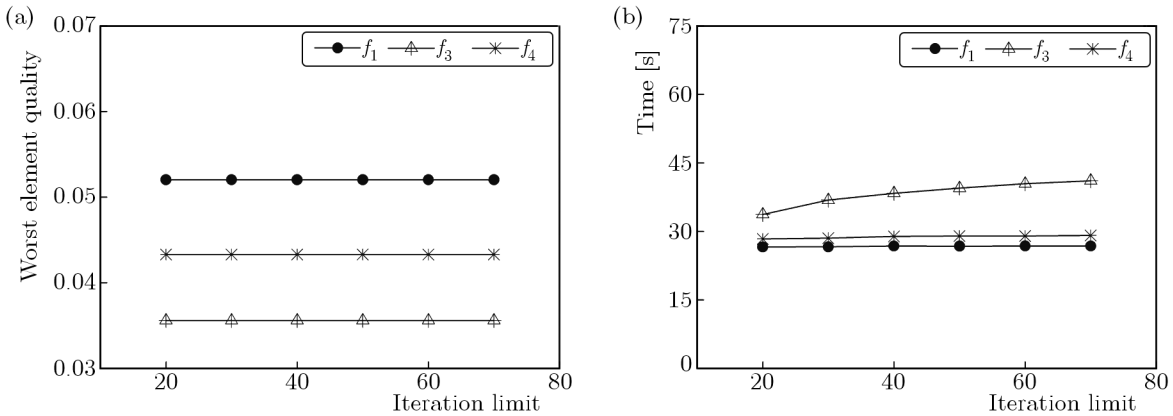

Fig. 9. Final mesh quality and time cost for different interaction limits

4.3. Effect of the desired accuracy

The research on the effect of desired accuracy on the optimization results is carried out on the third set of the mesh, too. The iteration limit N is set to 50. As shown in Fig. 10, it is observed that with the increment of the desired accuracy, the mesh quality and the time consumed increase. The first objective function has the greatest improvement over the other functions in the mesh quality. It is followed by f_4 and then f_3. For the time cost, the order is $f_1 < f_4 < f_3$.

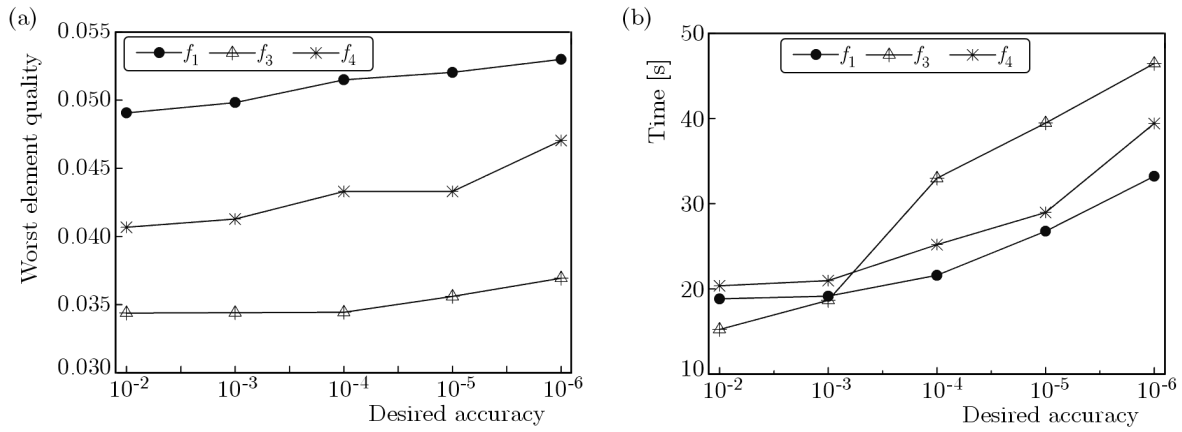

Fig. 10. Final mesh quality and time cost for different desired accuracy

4.4. Comparison with other methods

The objective function which is constructed by f_1 is better than other objective functions when the element number, iteration limit, and desired accuracy change. To evaluate our mesh improvement algorithm, comparison with Freitag and Ollivier-Gooch's method (Freitag and Ollivier-Gooch, 1997) whose objective function is deduced by minimum sine of the dihedral angles is made. And two cases of the mesh (TIRE and RAND2) come by courtesy of Freitag and Ollivier-Gooch. TIRE is a tire incinerator with 2 570 nodes and 11 099 tetrahedral elements. Its initial mesh quality can be found in Table 7. RAND2 are lazy triangulations generated by inserting randomly located nodes into a cube one by one. Each node is inserted by splitting one or more tetrahedra into multiple tetrahedra. The random meshes have horrible quality and poor dihedral angles at both extremes. The RAND2 mesh has 5 086 nodes and 25 704 tetrahedral elements; its initial quality can be found in Table 7. Table 7 compares the minimum and maximum dihedral angles reported by Freitag and Ollivier-Gooch to that achieved by our mesh improvement algorithm. It can be seen that the mesh quality is bad before the optimization, and there are badly shaped elements whose dihedral angles tend to be $0°$ or $180°$. Dihedral angles are improved to be between $2°$ and $178°$ for Freitag and Ollivier-Gooch's algorithm with smoothing, and between $10°$ and $156°$ for our proposed algorithm.

Table 7. Statistics of the examples before and after optimization

Model	Method	Before optimization		After optimization	
		Minimum dihedral angles	Maximum dihedral angles	Minimum dihedral angles	Maximum dihedral angles
TIRE	Freitag's	0.66°	178.88°	13.67°	161.71°
	proposed			15.27°	150.39°
RAND2	Freitag's	0.10°	179.84°	1.91°	177.69°
	proposed			10.31°	156.55°

The two mesh optimization algorithms have been implemented and tested for RAND2 with the distribution of dihedral angles, as shown in Fig. 11. Figure 11a shows the initial RAND2 mesh, and Figs. 11b and 11c show the optimized mesh by Freitag's algorithm and the proposed algorithm, respectively. It can be seen that they can both successfully eliminate poorly shaped elements from the mesh. Comparing with Freitag's algorithm, the proposed algorithm is more successful in eliminating poor dihedral angles at both extremes.

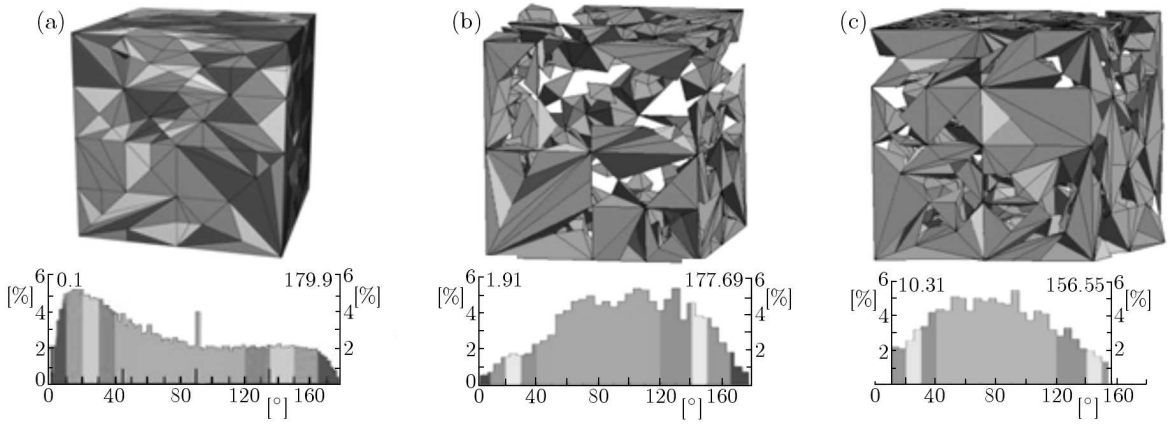

Fig. 11. Mesh quality improvement for RAND2 with two algorithms; (a) initial mesh, (b) optimized after Freitag's algorithm, (c) optimized after proposed algorithm

5. Conclusions

Seven shape metrics are compared by carrying out six numerical tests aiming at comparing their abilities to identify poor-quality elements and assessing the change of element shape. An optimization-based smoothing algorithm for tetrahedral mesh quality improvement is proposed, and the method used to calculate an optimal solution including the determination of the optimal search direction and step-size is studied.

The effects of element number, iteration limit and desired degree of accuracy on the performance of three different objective functions are assessed. The practical examples show that no matter which objective function is used in the proposed optimization-based smoothing method, the mesh quality can be significantly improved. With the increase of element number, the time consumed during optimization increases for the three objective functions. The iteration limit has little effect on the mesh quality and consuming time for different functions. The increasing of the desired degree of accuracy will improve the mesh quality and cost more time. For all the objective functions, when the element number, iteration limit and desired degree of accuracy change, the worst element quality will be improved in the following sequence, $f_3 < f_4 < f_1$. For the time of optimization, the arrangement is $f_1 < f_4 < f_3$. The results obtained with the approach for the objective function f_1 is compared with some common approach. It is clearly shown that it performs better than the existing approach.

Acknowledgements

This work was supported by National Natural Science Foundation of China (No. 51309119, 51109095,51179075), a project funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions, Natural Science Fund in Jiangsu Province (BK2010346, BK2009006), Postgraduate Innovation Foundation of Jiangsu Province (CXZZ12_0680) and the Advanced Talent Foundation of Jiangsu University (12JDG082).

References

1. Dobrzynski C., Frey P., 2008, Anisotropic Delaunay mesh adaptation for unsteady simulations, *Proceedings of the 17th International Meshing Roundtable*, 177-194

2. Dong L., Liu H.L., Tan M.G., Lu M.Z., Wang Y., Wang K., 2011, Quality measurement criteria and optimization algorithm of tetrahedral mesh for centrifugal pumps (in Chinese), *Journal of Xi'an Jiaotong University*, **45**, 11, 31-36

3. Edelsbrunner H., Shah N.R., 1996, Incremental topological flipping works for regular triangulations, *Algorithmica*, **15**, 3, 223-241

4. Freitag L.A., Ollivier-Gooch C., 1997, Tetrahedral mesh improvement using swapping and smoothing, *International Journal for Numerical Methods in Engineering*, **40**, 21, 3979-4002

5. Geuzaine C., Remacle J.-F., 2009, Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities, *International Journal for Numerical Methods in Engineering*, **79**, 11, 1309-1331

6. Hetmaniuk U., Knupp P., 2011, A mesh optimization algorithm to decrease the maximum interpolation error of linear triangular finite elements, *Engineering with Computers*, **27**, 1, 3-15

7. Klingner B.M., Shewchuk J.R., 2008, Aggressive tetrahedral mesh improvement, *Proceedings of the 17th International Meshing Roundtable*, 3-23

8. Lo S.H., 1997, Optimization of tetrahedral meshes based on element shape measures, *Computers and Structures*, **63**, 5, 951-961

9. Mao Z.H., Ma L.Z., Zhao M.X., Li Z., 2006, A modified Laplacian smoothing approach with mesh saliency, *Smart Graphics*, **4073**, 105-113

10. Montenegro R., Cascón J.M., Escobar J.M., Rodríguez E., Montero G., 2009, An automatic strategy for adaptive tetrahedral mesh generation, A*pplied Numerical Mathematics*, **59**, 9, 2203-2217

11. Munson T., 2007, Mesh shape-quality optimization using the inverse mean-ratio metric, *Mathematical Programming*, **110**, 3, 561-590

12. Nie C.H., 2003, Study on quality measures for tetrahedral mesh (in Chinese), *Chinese Journal of Computational Mechanics*, **20**, 5, 579-582

13. Park J., Shontz S.M., 2010, Two derivative-free optimization algorithms for mesh quality improvement, *Procedia Computer Science*, **1**, 1, 387-396

14. Parthasarathy V.N., Graichen C.M., Hathaway A.F., 1994, A comparison of tetrahedron quality measures, *Finite Elements in Analysis and Design*, **15**, 3, 255-261

15. Shewchuk J.R., 2002, Delaunay refinement algorithms for triangular mesh generation, *Computational Geometry*, **22**, 1/3, 21-74

16. Si H., Gärtner K., 2011, 3D boundary recovery by constrained Delaunay tetrahedralization, *International Journal for Numerical Methods in Engineering*, **85**, 11, 1341-1364

17. Tournois J., Wormser C., Alliez P., Desbrun M., 2009, Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation, *ACM Transactions on Graphics*, **28**, 3, 1557-1571

18. Vartziotis D., Wipper J., Schwald B., 2009, The geometric element transformation method for tetrahedral mesh smoothing, *Computer Methods in Applied Mechanics and Engineering*, **199**, 1/4, 169-182

19. Xu K., Cheng Z.Q., Wang Y.Z., Xiong Y., Zhang H., 2009, Quality encoding for tetrahedral mesh optimization, *Computers and Graphics*, **33**, 3, 250-261

Manuscript received January 15, 2013; accepted for print August 2, 2013