Automatic adaptive mesh generation using metric advancing front approach

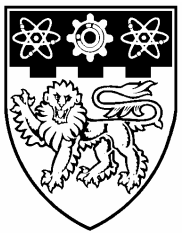
Lee, Chi King

1999

Lee, C. (1999). Automatic adaptive mesh generation using metric advancing front approach. Engineering Computations, 16(2), 230-263.

https://hdl.handle.net/10356/103384

https://doi.org/10.1108/02644409910257494



Automatic Adaptive Mesh Generation Using Metric Advancing Front Approach

C. K. Lee

School of Civil and Structural Engineering
Nanyang Technological University
Nanyang Avenue, Singapore 639798
Email: ccklee@ntu.edu.sg

Pre-printed version for the paper appeared in the journal
Engineering Computations, Vol. 16, 1999, 230-263

Summary

A new mesh generation procedure is suggested for the generation of 2D adaptive finite element meshes with strong element gradation and stretching effects. Metric tensors are employed to define and control the element characteristics during the mesh generation process. By using the metric tensor specification and a new, robust and refined advancing front triangulation kernel, triangles with nearly unit edge length with respect to the normalized space are generated. Highly graded and stretched elements can be generated without much difficulty and the operation complexity of the mesh generation process is exactly the same as the usual 2D advancing front mesh generator. A set of mesh quality enhancement procedures has also been suggested for the further improvement of the quality of the finite element meshes. A simple and effective mesh conversion scheme is used to convert the output triangular mesh to a pure quadrilateral mesh while all the essential element characteristics are preserved. Mesh generation examples show that high quality finite element meshes with element characteristics compatible with the specified metric tensors are generated within a reasonable time limit in a common small computing environment.

Keywords: Automatic mesh generation, advancing front technique, Metric specification, stretching ratio and directions

1. Introduction

When applying the finite element method to practical applications, one of the most time consuming steps is the creation of a proper discretization model for the problem domain. For an application in which a complicated domain is involved, it is quite often that the time spent on the creation of the finite element mesh is longer than the time used in the finite element analysis[1]. Furthermore, with the rapid advancement in the areas of error analysis and adaptivity[2-5], the role of a fully automatic mesh generator becomes increasingly important. Consequently, much effort has been devoted to the development of highly effective and robust automatic mesh generation algorithms. It appears that among the many methods available to date[6,7], the following two approaches are the most effective, robust and reliable to generate high quality meshes.

(1) The structural partition/refinement technique

This generally refers to those recursive domain decomposition methods[8,9] and the classical Delaunary-Voronoi approach[10,11]. In the recursive domain decomposition methods, the domain to be meshed is first repetitive broken down into simpler units until standard templates can be used to discretize them. In the Delaunary approach, a coarse mesh is usually first generated to cover the problem domain. This coarse mesh is then systemically refined by repetitive insertions of nodes and by the modifications of the element connectivity until the required gradation effect is achieved.

(2) The advancing front technique (AFT)[12,13]

In this approach, the boundary of the problem domain will always be discretized first and then nodes and elements are generated one by one to fill up the problem domain in a hierarchical manner.

It should be pointed out that in many applications[5,14,15], it is frequently required that the mesh generator should able to generate finite element meshes with element characteristics (grading, stretching factors and directions) satisfying the stringent requirements prescribed by the user or the adaptive mesh refinement scheme. Hence, in addition to devise a robust element formation algorithm, the precise representation and control of element characteristics are also important in automatic mesh generation. Initially, the required characteristics of the mesh were manually defined by the users in terms of a few parameters[16,17]. As more and more sophisticated mesh generation algorithms were developed, more versatile methods using the *node spacing function* (NSF) approach to specify the element size and grading requirements over the entire problem domain through finite element nodal interpolation were

adopted[6,12,18]. Driven by practical requirements, a more complicated description of the element characteristics is frequently needed and as a result many more complex forms of NSF which include various element distortion effects were used[14,19,20]. Very recently, H. Borouchaki et al[11,21,22] devised a general *Delaunary* mesh generator based on the *metric specification* approach in which all the element size characteristics were represented by a single metric map during mesh generation. The metric map was defined over the entire problem domain by the NSF approach.

The main objective of this paper is to suggest a new 2D automatic mesh generator based on the *advancing front technique* using the metric specification for element characteristics control. It will be shown that all essential generation and mesh quality enhancement steps of a robust advance front mesh generator can easily be modified and used in conjunction with the metric specification approach. The resulting metric advancing front mesh generator will be capable to generate meshes satisfying the user specified gradation and directional stretching effects. In the next section, a concise summary of all the essential definitions used in the metric specification will be given. It will then be followed by the detailed description of the new metric advancing front mesh generator and all the *ad hoc* mesh quality enhancement steps developed. Finally, several mesh generation examples will be given to demonstrate the robustness and effectiveness of the new mesh generator.

2. The metric specifications

In this section, a concise summary of the metric specification is given in order to make clear the subsequently descriptions of the new mesh generator. Details of the metric approach can be found in references [11] and [21].

*(1) The metric tensor **M** and length scale definition*

Let Ω be the problem domain of $\Re^n$ to be meshed. For any points $\mathbf{P}_1$ and $\mathbf{P}_2$ in Ω, consider the n×n symmetric positive definite matrix (or the metric tensor) $\mathbf{M}$ and the real function, $\tilde{l}(\mathbf{M}, \mathbf{P}_1\mathbf{P}_2)$, defined as

$$\tilde{l}(\mathbf{M}, \mathbf{P}_1\mathbf{P}_2) = \int_0^1 \sqrt{\mathbf{Q}(\mathbf{P}_1, \mathbf{P}_2, t)^T \mathbf{M}(\mathbf{Q}(\mathbf{P}_1, \mathbf{P}_2, t))\mathbf{Q}(\mathbf{P}_1, \mathbf{P}_2, t)}\,dt \qquad (1a)$$

where

$$\mathbf{Q}(\mathbf{P}_1, \mathbf{P}_2, t) = \mathbf{P}_1 + t(\mathbf{P}_2 - \mathbf{P}_1) \quad 0 \le t \le 1 \qquad (1b)$$

Then one can verify that $\tilde{l}(\mathbf{M}, \mathbf{P}_1\mathbf{P}_2)$ is in fact a metric[23] defined over Ω. Obviously, if $\mathbf{M}=\mathbf{I}_n$, the n×n identity matrix, then $\tilde{l}(\mathbf{M}, \mathbf{P}_1\mathbf{P}_2)$ will be the usual length of the vector $\mathbf{P}_2$-$\mathbf{P}_1$ (or the distance between the two points $\mathbf{P}_1$ and $\mathbf{P}_2$) in $\Re^n$. In general, $\tilde{l}(\mathbf{M}, \mathbf{P}_1\mathbf{P}_2)$ is called the length of $\mathbf{P}_2$-$\mathbf{P}_1$ (or distance between $\mathbf{P}_1$ and $\mathbf{P}_2$) *with respect to the metric tensor* **M**.

The significance of the metric tensor **M** can be seen by considering the particular case that n=2 so that the matrix **M** is given by

$$\mathbf{M} = \begin{bmatrix} a & b \\ b & c \end{bmatrix} \tag{2a}$$

Since **M** is symmetric and positive definite, it can always be expressed in terms of its eigenvalues, λ_1 and $\lambda_2 > 0$ and their corresponding eigenvectors, $\mathbf{e}_1$ and $\mathbf{e}_2$. That is

$$\mathbf{M} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix}^T \tag{2b}$$

Since λ_i, i=1,2 > 0, one can write $\lambda_i = 1/(h_i)^2$. For instance, assume that **M** is constant and by using Eqns. 1 and 2 and the orthogonal property of $\mathbf{e}_i$. The length of a vector $\mathbf{u}=\mathbf{P}_2$-$\mathbf{P}_1$ in the $\mathbf{e}_1$ (respectively the $\mathbf{e}_2$) direction with length equal to h_1 (respectively h_2) will then has unit length with respect to **M**. That is

$$\tilde{l}(\mathbf{M}, \mathbf{P}_1\mathbf{P}_2) = \tilde{l}(\mathbf{M}, \mathbf{u}) = 1 \tag{3}$$

In any other direction, the length of the vector will be given by the ellipse defined by **M** (Fig. 1). Therefore, the metric tensor **M** that defines the length scale measure can be used to control the element characteristics of the mesh. The element size and stretch ratio are defined by the parameter h_i (hence λ_i). When $h_1=h_2$ the metric is *isotropic* and in such a case the directions $\mathbf{e}_1$ and $\mathbf{e}_2$ are immaterial. Otherwise, the metric is *anisotropic* so that $\mathbf{e}_1$ and $\mathbf{e}_2$ will define the stretching directions of the elements in the mesh. With the above definition of length transformation by the metric tensor, the task of generating a finite element mesh with prescribed element characteristics will simply become the problem of generating elements with *unit edge length* with respect to the prescribed metric tensor. Hence, during mesh generation, a perfect triangle will be a unit equilateral triangle with respect to the metric tensor while a unit square will be a prefect quadrilateral element.

Note that the above interpretation of the metric can easily be generalized to both 3D and surface mesh generations. Details of these are given in Appendix A.

(2) Metric tensor interpolation along a line segment

Suppose that $\mathbf{M}_1$ and $\mathbf{M}_2$ are the metric tensors defined at the two end points of the line $\mathbf{P}_1\mathbf{P}_2$ respectively. Then the metric tensor $\mathbf{M}$ defined at the point $\mathbf{P}=\mathbf{P}_1+t(\mathbf{P}_2-\mathbf{P}_1)$ can be obtained by the simultaneous matrix reduction method[22]. A concise summary of the method is given below.

(i) Solve the generalized eigenvalues problem

$$\mathbf{M}_1\mathbf{S}=\mathbf{\Lambda}\mathbf{M}_2\mathbf{S} \tag{4}$$

where $\mathbf{S}=\begin{bmatrix}\mathbf{s}_1 & \mathbf{s}_2\end{bmatrix}$ is the eigenvector matrix for Eqn. 4 and $\mathbf{\Lambda}$ is the corresponding eigenvalue matrix.

(ii) Compute the lengths r_i and w_i, i=1,2

$$r_i=\frac{1}{\sqrt{\mathbf{s}_i^T\mathbf{M}_1\mathbf{s}_i}} \quad \text{and} \quad w_i=\frac{1}{\sqrt{\mathbf{s}_i^T\mathbf{M}_2\mathbf{s}_i}} \tag{5}$$

(iii) Interpolation of the principal length, h_i, i=1,2 at $\mathbf{P}$

$$h_i(t)=r_i+t(r_i-w_i) \tag{6}$$

(iv) Construction of $\mathbf{M}$ by

$$\mathbf{M}=\left(\mathbf{S}^{-1}\right)^T\begin{bmatrix}1/\left(h_1(t)\right)^2 & 0\\ 0 & 1/\left(h_2(t)\right)^2\end{bmatrix}\mathbf{S}^{-1} \tag{7}$$

(3) Metric interpolation from background mesh

In general, the metric tensor varies over the problem domain and the NSF approach is employed to define it implicitly at the nodal points of the background mesh[14]. During mesh generation, one frequently required operation is the interpolation of the metric tensor at a given point on the current mesh generation front from the background mesh. In the current implementation, the *Alternated Digital Tree* technique[24] is used to cut down the computational cost for locating the background element containing the point. Once the element contains the point is located the local co-ordinates of the point $\mathbf{P}$ with respect to that element can then be computed for interpolation. For triangular element, interpolation along one parent co-ordinate direction will first be carried out to obtain the metric tensor at the point $\mathbf{Q}_1$ (Fig. 2a). If the element is a quadrilateral, interpolations will first be carried out at the points $\mathbf{Q}_1$ and $\mathbf{Q}_2$ (Fig. 2b). The metric tensor at $\mathbf{P}$ is then obtained by a second interpolation (along the line $\mathbf{P}_1\mathbf{Q}_1$ for triangle and the line $\mathbf{Q}_1\mathbf{Q}_2$ for quadrilateral).

(4) Metric intersection

In many applications, it is necessary to use more than one metric tensor to describe the characteristics of the target mesh. In such a situation, the simultaneous matrix reduction method can be used again to compute the combined effect of two metric tensors. If $\mathbf{M}_1$ and $\mathbf{M}_2$ are two metric tensors, the combined metric tensor, $\mathbf{M}$, can be defined as

$$\mathbf{M} = \left(\mathbf{S}^{-1}\right)^{\mathsf{T}} \begin{bmatrix} \max(\lambda_1, \mu_1) & 0 \\ 0 & \max(\lambda_2, \mu_2) \end{bmatrix} \mathbf{S}^{-1} \tag{8}$$

where $\mathbf{S}$ is the eigenvector matrix of the generalized eigenvalue problems stated in Eqn. 4 while λ_i and μ_i, i=1,2, are respectively the eigenvalues of $\mathbf{M}_1$ and $\mathbf{M}_2$ respectively. Obviously, in case that more than two metric maps are present, Eqn. 8 can be applied recursively.

(5) Angle measure

In addition to the length scale measure, angle measure can also be made with respect to the metric tensor. The angle between two elementary (infinitesimal) vectors $\mathbf{du}$ and $\mathbf{dv}$ with respect to the metric tensor $\mathbf{M}$, $\tilde{\theta}(\mathbf{M}, \mathbf{du}, \mathbf{dv})$, is defined as

$$\tilde{\theta}(\mathbf{M}, \mathbf{du}, \mathbf{dv}) = \begin{cases} \cos^{-1}\left(\dfrac{(\mathbf{du})^{\mathsf{T}} \mathbf{M} \mathbf{dv}}{\tilde{l}(\mathbf{M}, \mathbf{du}) \cdot \tilde{l}(\mathbf{M}, \mathbf{dv})} \right) & \text{if } \mathbf{du} \times \mathbf{dv} > 0 \\ 2\pi - \cos^{-1}\left(\dfrac{(\mathbf{du})^{\mathsf{T}} \mathbf{M} \mathbf{dv}}{\tilde{l}(\mathbf{M}, \mathbf{du}) \cdot \tilde{l}(\mathbf{M}, \mathbf{dv})} \right) & \text{otherwise} \end{cases} \tag{9}$$

(6) Normalized vector space and rotation of infinitesimal vector

Another useful interpretation of the metric tensor $\mathbf{M}$ is to consider its square root, $\mathbf{M}^{1/2}$, as a transformation matrix so that the length of an infinitesimal vector $\mathbf{du}$ with respect to $\mathbf{M}$ is nothing more than the length of the transformed vector $\mathbf{d\xi}$[20] defined in a normalized vector space. That is

$$\tilde{l}(\mathbf{M}, \mathbf{du})^2 = \mathrm{d}\xi^{\mathsf{T}} \mathrm{d}\xi = \left(\mathbf{M}^{1/2} \mathbf{du}\right)^{\mathsf{T}} \left(\mathbf{M}^{1/2} \mathbf{du}\right) = \mathbf{du}^{\mathsf{T}} \mathbf{M} \mathbf{du} \tag{10a}$$

From Eqn. 10a, the transformed vector $\mathbf{d\xi}$ is defined as

$$\mathbf{d\xi} = \mathbf{M}^{1/2} \mathbf{du} \tag{10b}$$

Following this, the elementary vector $\mathbf{d\zeta}$ formed by rotating the elementary vector $\mathbf{d\xi}$ through an anti-clockwise angle φ can be computed as

$$\mathbf{d\zeta} = \mathbf{R}\mathbf{d\xi} \tag{11}$$

where **R** is the rotational matrix defined as

$$\mathbf{R}(\varphi) = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \tag{12}$$

As a result, the vector **dv** corresponding to **dζ** can be obtained by

$$\mathbf{M}^{1/2}\mathbf{dv} = \mathbf{d\zeta} = \mathbf{R}(\varphi)\mathbf{d\xi} = \mathbf{R}(\varphi)\cdot\mathbf{M}^{1/2}\mathbf{du} \quad \text{or} \quad \mathbf{dv} = \mathbf{M}^{-1/2}\cdot\mathbf{R}(\varphi)\cdot\mathbf{M}^{1/2}\mathbf{du} \tag{13}$$

For two-dimensional applications in which the metric tensor is given by Eqn. 2. The matrices $\mathbf{M}^{1/2}$ and $\mathbf{M}^{-1/2}$ can be expressed as

$$\mathbf{M}^{1/2} = \frac{1}{\sqrt{a+c+2d}}\begin{bmatrix} a+d & b \\ b & c+d \end{bmatrix} \tag{14a}$$

$$\mathbf{M}^{-1/2} = \frac{1}{d\sqrt{a+c+2d}}\begin{bmatrix} c+d & -b \\ -b & a+d \end{bmatrix} \tag{14b}$$

where $d = \sqrt{ac - b^2}$.

(7) Shape quality of elements with respect to the metric tensor

One characteristic of an advancing front mesh generator is that during each element formation step, it is aimed that the element generated is always the best in terms of the shape and grading with respect to the prescribed specifications. Hence, a quantitative measure for the quality of element is necessary. The shape quality for a triangle (respectively a quadrilateral) can be measured by computing the α value (respectively the β value)[25] of the element with respect to the metric tensor and is denoted as $\tilde{\alpha}$ (respectively $\tilde{\beta}$). For the triangle shown in Fig. 3, its $\tilde{\alpha}$ value is given by

$$\tilde{\alpha} = \min(\hat{\alpha}(\mathbf{P}_1), \hat{\alpha}(\mathbf{P}_2), \hat{\alpha}(\mathbf{P}_3)) \tag{15a}$$

where

$$\hat{\alpha}(\mathbf{P}_i) = \frac{2\sqrt{3}\cdot \mathrm{Det}(\mathbf{M}_i)\cdot \mathrm{Det}([\mathbf{P}_2 - \mathbf{P}_1, \ \mathbf{P}_3 - \mathbf{P}_1])}{\tilde{l}(\mathbf{M}_i, \mathbf{P}_1\mathbf{P}_2)^2 + \tilde{l}(\mathbf{M}_i, \mathbf{P}_2\mathbf{P}_3)^2 + \tilde{l}(\mathbf{M}_i, \mathbf{P}_1\mathbf{P}_3)^2} \tag{15b}$$

and Det(**M**) is the determinant of the matrix **M**.

For the quadrilateral shown in Fig. 3, the four $\tilde{\alpha}$ values, $\tilde{\alpha}_i$ i=1,4, corresponding to the four triangles formed by using the two diagonals $\mathbf{P}_1\mathbf{P}_3$ and $\mathbf{P}_2\mathbf{P}_4$ are first computed. They are then arranged in the order

$$\tilde{\alpha}_1 \geq \tilde{\alpha}_2 \geq \tilde{\alpha}_3 \geq \tilde{\alpha}_4 \tag{16a}$$

The $\tilde{\beta}$ quality is then defined as

$$\tilde{\beta} = \frac{\tilde{\alpha}_3 \tilde{\alpha}_4}{\tilde{\alpha}_1 \tilde{\alpha}_2} \tag{16b}$$

(8) Combined shape and size measures

Note that the $\tilde{\alpha}$ and $\tilde{\beta}$ values defined in Eqns. 15 and 16 only account for the shape distortion effect of the element with respect to the metric tensor. An equilateral triangle with edge length equal to 2.0 will also attain a maximum $\tilde{\alpha}$ value of 1.0. Hence, during the element formation step, a combined shape and size measure will be used to assess the overall quality of the element[26]. This can be done by using the $\tilde{\mu}$ quality parameter defined as follow

$$\tilde{\mu} = \begin{cases} \tilde{\alpha} e^{-\delta} & \text{for triangles} \\ \tilde{\beta} e^{-\delta} & \text{for quadrilaterals} \end{cases} \tag{17a}$$

where δ is the average derivation of the edge length of the element from unity and is defined as

$$\delta = \frac{1}{N_V} \sum_{i=1}^{N_V} \left| \tilde{l}\,(\mathbf{M}_i, \mathbf{P}_i \mathbf{P}_j) - 1 \right|$$
$$N_V = \text{Number of edges of the element and } j = \begin{cases} i+1 & \text{if } i \neq N_V \\ 1 & \text{if } i = N_V \end{cases} \tag{17b}$$

so that elements with edge length deviated from unity will have a low $\tilde{\mu}$ value. For example, for an equilateral triangle with edge length equal to 2.0, the $\tilde{\mu}$ value of the element will equal to 0.367. Therefore, the $\tilde{\mu}$ value can be used as the criterion for the selection of the best element during the element formation step.

3. Metric advancing front mesh generation

Overview

Similar to many traditional advancing front mesh generators, the metric advancing front mesh generator presented here also generates a finite element mesh in a hierarchic manner. Boundary segments will first be generated and then followed by the main triangulation process. Mesh enhancement procedures will then be employed to improve the quality of the mesh. Finally, if a quadrilateral mesh is required, a simple algorithm will convert the output triangular mesh to a pure quadrilateral mesh.

Details of these generation steps will be given in the following four sections.

Boundary segments generation

In general, a number of boundary curves are used to provide the topological and geometrical descriptions of the boundary of the problem domain. (In the current implementation, each boundary curve is modelled by a non-uniform rational spline curve.) Therefore, each boundary curve must first be decomposed into a number of unit length boundary segments (with respect to the prescribed metric tensor) for the formation of the initial generation front. This can be accomplished by the following steps.

(i) Computation of the total length of the old boundary segments.

The total length of the old boundary segments, $\widetilde{L}$ (Fig. 4), of a given boundary curve can be easily obtained by simple summation. i.e.

$$\widetilde{L} = \sum_{i=1}^{N_O} \widetilde{l}\,(\mathbf{M}, \mathbf{P}_i\mathbf{P}_{i+1}) \tag{18}$$

where N_O is the number of old segments of the curve. Note that, in general, the metric tensor $\mathbf{M}$ varies along the segment $\mathbf{P}_i\mathbf{P}_{i+1}$ and Eqn. 18 can only be evaluated numerically. In the current implementation, the length of a segment is evaluated by the subdivision algorithm described in Appendix B.

(ii) Computation of the number of new segments

Once $\widetilde{L}$ is computed, the number of new segments for this curve, N_G, will be determined as

$$N_G = \begin{cases} m & \text{if } \dfrac{m}{\widetilde{L}} > \dfrac{\widetilde{L}}{m+1} \\ m+1 & \text{otherwise} \end{cases} \tag{19}$$

where m is an integer such that $m \le \tilde{L} \le m+1$.

(iii) Formation of new boundary nodes

Let $d=\tilde{L}/N_G$, then new boundary nodes $\mathbf{Q}_i$, i=1,....,N_G+1 will be created in such a way that the total running length up to the node $\mathbf{Q}_i$ will equal to (i-1)·d. This can be done by first locating the jth existing segment such that

$$\sum_{k=1}^{j-1} \tilde{l}(\mathbf{M}, \mathbf{P}_k \mathbf{P}_{k+1}) \le (i-1) \cdot d \le \sum_{k=1}^{j} \tilde{l}(\mathbf{M}, \mathbf{P}_k \mathbf{P}_{k+1}) \qquad (20)$$

After this segment is found, the position of the new point $\mathbf{Q}_i$ will be obtained by simple linear interpolation.

(iv) Computation of parametric values of new nodes on the boundary curve

The parametric values of the nodes generated with respect to the non-uniform rational B-spline curves are then interpolated (again linearly) from the parametric values of the old boundary segments.

Metric Triangulation

Once all boundary curves are discretized, the initial generation front can be established as usual. A pure triangular mesh will be formed by using the advancing front technique. The major difference between the new metric mesh generation procedure and a traditional advancing front mesh generator is the element formation step. In metric mesh generation, the size and all other geometrical characteristics of the elements are always referred to the metric map. It is always aimed to generate unit equilateral triangles with respect to the metric map. All other topological checks (e.g. determination of a valid front node for element formation) will be carried out in exactly the same way as in a traditional advancing front mesh generator. The element formation step starts with the selection of the shortest front segment (in the physical space) in the current generation front as the base segment. A new element will be generated at the base segment either by creating a new interior node or by connecting it to an existing front node. The choice between a new interior node and an existing boundary node will depend on the $\tilde{\mu}$ quality (Eqn. 17) of the resulting element. After an element is formed, the generation front will be updated as usual and the generation process is terminated when the number of front segments is reduced to zero. The new element formation procedure consists of the following steps.

(i) The shortest segment **AB** (Fig. 5) is selected as the base segment from the current generation front.

(ii) Compute the length $\tilde{l}(\mathbf{M},\mathbf{AB})$ and locate the pseudo-midpoint **D** on **AB** (using the method described in Appendix B) such that

$$\tilde{l}(\mathbf{M},\mathbf{AD}) \approx \tilde{l}(\mathbf{M},\mathbf{DB}) \approx 0.5 \tag{21}$$

Also, the metric tensors at points **A**, **B** and **D** will be retrieved from the background mesh. They are denoted as $\mathbf{M_A}$, $\mathbf{M_B}$ and $\mathbf{M_D}$ respectively.

(iii) Establish the best offset point **C** in such a way that the triangle **ABC** will be as close to a unit equilateral triangle as possible in the normalized space. Hence, it is required that point **C** must satisfy the conditions

$$\begin{aligned} &\tilde{l}(\mathbf{M},\mathbf{AC}) = \tilde{l}(\mathbf{M},\mathbf{BC}) = 1,\ \tilde{l}(\mathbf{M},\mathbf{DC}) = \sqrt{3}/2 \\ &\tilde{\theta}(\mathbf{M},\mathbf{AB},\mathbf{AC}) = \tilde{\theta}(\mathbf{M},\mathbf{BC},\mathbf{BA}) = \pi/3,\ \tilde{\theta}(\mathbf{M},\mathbf{DB},\mathbf{DC}) = \pi/2 \end{aligned} \tag{22}$$

Since, in general, the metric tensor **M** varies over the problem domain, conditions stated in Eqn. 22 can only be satisfied approximately. A good trial position of **C**, denoted as **C1**=$(u_{C1},v_{C1})^T$, can be found by rotating the vector **AB** at point **A** for an angle equal to $\pi/3$ in the normalized space. By using Eqns. 12 and 13, $(u_{C1},v_{C1})^T$ can be expressed as

$$\begin{pmatrix} u_{C1} \\ v_{C1} \end{pmatrix} = \begin{pmatrix} u_A \\ v_A \end{pmatrix} + \frac{1}{\tilde{l}(\mathbf{M_A},\mathbf{AB})} \mathbf{M_A}^{-1/2} \mathbf{R}(\frac{\pi}{3}) \mathbf{M_A}^{1/2} \begin{pmatrix} u_B - u_A \\ v_B - v_A \end{pmatrix} \tag{23a}$$

If $\mathbf{M_A} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$, then by using Eqn. 14, Eqn. 23a can be written as

$$\begin{pmatrix} u_{C1} \\ v_{C1} \end{pmatrix} = \begin{pmatrix} u_A \\ v_A \end{pmatrix} + \frac{\sqrt{3}}{2d \cdot \tilde{l}(\mathbf{M_A},\mathbf{AB})} \begin{pmatrix} \frac{d}{\sqrt{3}} - b & -c \\ a & \frac{d}{\sqrt{3}} + b \end{pmatrix} \begin{pmatrix} u_B - u_A \\ v_B - v_A \end{pmatrix}, \quad d = \sqrt{ac - b^2} \tag{23b}$$

Similarly, two other possible positions of **C**, denoted as **C2**=$(u_{C2},v_{C2})^T$ and **C3**=$(u_{C3},v_{C3})^T$, can be established by rotating the vectors **BA** at point **B** for an angle equal to $-\pi/3$ and the vector **DB** at point **D** for an angle equal to $\pi/2$ respectively. The co-ordinates of **C2** and **C3** are given in Appendix C. After these three points are established, the $\tilde{\mu}$ values corresponding to the triangles formed by these three points with the base segment **AB** will be calculated. The best offset point **C**=$(u_C,v_C)^T$ will then be taken as the one corresponding to the highest $\tilde{\mu}$ value.

(iv) The searching ellipse ***E*** and the valid node set **Ψ**

After the best offset point **C** is established, the metric tensor at **C**, $\mathbf{M_C}$, is retrieved from the background mesh and the length $\tilde{l}(\mathbf{M_C}, \mathbf{DC})$ will be computed. The eigenpairs of $\mathbf{M_C}$, $(\{\lambda_1^{\mathbf{C}} = 1/(h_1^{\mathbf{C}})^2, \lambda_2^{\mathbf{C}} = 1/(h_2^{\mathbf{C}})^2\}, \{\mathbf{e}_1^{\mathbf{C}}, \mathbf{e}_2^{\mathbf{C}}\})$ are then computed to form the searching ellipse ***E*** with centre at **C** (Fig. 5). Note that in order to limit the final size of the elements formed, the principal radii of ***E*** are limited to $0.75h_1^{\mathbf{C}}$ and $0.75h_2^{\mathbf{C}}$. All the front nodes that are lying inside ***E*** which will form a valid element with the base segment **AB** are then collected into a valid node list denoted as **Ψ**. Point **C** will also be added into **Ψ** if it is also a valid node with respect to **AB** and the shortest distance (with respect to $\mathbf{M_C}$) between **C** and all current front segments except **AB** is greater than or equal to 0.1.

(v) Selection of the third point of the element

If **Ψ** is not empty, then the node which results in an element with the highest $\tilde{\mu}$ value will be selected as third node, **F**, for element formation. The element formation process is then finished and goto step (vii).

Otherwise, the offset point **C** will be moved toward **D** along the line **CD** such that

$$\begin{pmatrix} u_{\mathbf{C}} \\ v_{\mathbf{C}} \end{pmatrix} \leftarrow \begin{pmatrix} u_{\mathbf{D}} \\ v_{\mathbf{D}} \end{pmatrix} + 0.8 \begin{pmatrix} u_{\mathbf{C}} - u_{\mathbf{D}} \\ v_{\mathbf{C}} - v_{\mathbf{D}} \end{pmatrix} \tag{24}$$

The metric tensor $\mathbf{M_C}$ and the length $\tilde{l}(\mathbf{M_C}, \mathbf{DC})$ are then recalculated.

(vi) If $\tilde{l}(\mathbf{M_C}, \mathbf{DC}) \geq 0.1$, steps (iv) and (v) will be repeated to update the valid node set **Ψ** and try to establish the third node **F** again. Otherwise, the best valid front node (node that will lead to a maximum $\tilde{\mu}$ value) on the current mesh generation front will be taken as the third node **F**.

(vii) The new element **ABF** is formed and the generation front will be updated.

Steps (i) to (vii) will be repeated until the whole problem domain is triangulated.

Mesh quality enhancement

After the triangulation procedure is finished, the quality of the finite element mesh will be improved by a series of mesh quality enhancement procedures involving swapping and smoothing.

(1) Diagonal swapping

This is a simple procedure to improve the quality of the mesh, all diagonals shared by two adjacent elements will be examined (Fig. 6). Diagonal **AC** will be swapped to **BD** if the product of the $\tilde{\beta}$ qualities of the resulting elements is greater than that before swapping.

(2) Edges length smoothing

This smoothing procedure is employed to adjust the positions of the nodes in such a way that the distance between two connected nodes will as close to unity as possible. Let node **P** be the node under consideration, all the nodes connected to it will be identified (Fig. 7). For the node $\mathbf{Q}_i$ the new ideal position of **P**, denoted as $\mathbf{P}'$, such that $\tilde{l}(\mathbf{M}, \mathbf{Q}_i\mathbf{P}') = 1$ can be obtained by

$$\mathbf{P}' = \mathbf{Q}_i + \frac{1}{\tilde{l}(\mathbf{M}, \mathbf{Q}_i\mathbf{P})}(\mathbf{Q}_i - \mathbf{P}) \tag{25a}$$

Similarly, if quadrilaterals are present, the ideal position with respect to the node $\mathbf{T}_i$ will be

$$\mathbf{P}' = \mathbf{T}_i + \frac{\sqrt{2}}{\tilde{l}(\mathbf{M}, \mathbf{T}_i\mathbf{P})}(\mathbf{T}_i - \mathbf{P}) \tag{25b}$$

Note that in Eqn. 25, $\tilde{l}(\mathbf{M}, \mathbf{Q}_i\mathbf{P})$ and $\tilde{l}(\mathbf{M}, \mathbf{T}_i\mathbf{P})$ are calculated using the method described in Appendix B. The new position of **P** is then defined as the centroid of all the positions of $\mathbf{P}'$ obtained from all the connected nodes. This new position will be accepted if and only if the product of all the $\tilde{\alpha}$ and $\tilde{\beta}$ values of the surrounding elements is increased after the reposition. Usually, no more than three cycles of length smoothing will be sufficient to achieve the convergence.

(3) Elements quality smoothing

This is employed to improve the quality of the elements surrounding a node. Again, consider the point **P** shown in Fig. 7. For the triangle $\mathbf{PG}_i\mathbf{H}_i$, the ideal position $\mathbf{P}'$ can be obtained by first normalizing and then rotating the edge $\mathbf{G}_i\mathbf{H}_i$ at the point $\mathbf{G}_i$ for an angle equal to $\pi/3$. That is

$$\mathbf{P}' = \mathbf{G}_i + \frac{1}{\tilde{l}(\mathbf{M}, \mathbf{G}_i\mathbf{H}_i)}\mathbf{M}_{\mathbf{G}_i}^{-1/2}\mathbf{R}(\frac{\pi}{3})\mathbf{M}_{\mathbf{G}_i}^{1/2}(\mathbf{H}_i - \mathbf{G}_i) \tag{26}$$

where $\mathbf{M}_{\mathbf{G}_i}$ is the metric tensor at $\mathbf{G}_i$.

For the quadrilateral $\mathbf{PQ}_i\mathbf{T}_i\mathbf{G}_i$, the ideal point is established by averaging the positions of the two points obtained by normalizing and rotating the edges $\mathbf{Q}_i\mathbf{T}_i$ and $\mathbf{G}_i\mathbf{T}_i$ for an angle equal to $\pi/2$ and $-\pi/2$ respectively. That is

$$\mathbf{P}' = \frac{1}{2}(\mathbf{P}_1' + \mathbf{P}_2') \tag{27a}$$

where

$$\mathbf{P}_1' = \mathbf{Q}_i + \frac{1}{\tilde{l}(\mathbf{M}, \mathbf{Q}_i\mathbf{T}_i)}\mathbf{M}_{\mathbf{Q}_i}^{-1/2}\mathbf{R}(\frac{\pi}{2})\mathbf{M}_{\mathbf{Q}_i}^{1/2}(\mathbf{T}_i - \mathbf{Q}_i) \tag{27b}$$

and

$$\mathbf{P}_2' = \mathbf{G}_i + \frac{1}{\tilde{l}(\mathbf{M}, \mathbf{G}_i\mathbf{T}_i)}\mathbf{M}_{\mathbf{G}_i}^{-1/2}\mathbf{R}(-\frac{\pi}{2})\mathbf{M}_{\mathbf{G}_i}^{1/2}(\mathbf{T}_i - \mathbf{G}_i) \tag{27c}$$

The new position of $\mathbf{P}$ is then defined as the centroid of all the positions of $\mathbf{P}'$ obtained from Eqns. 26 and 27. Again, the point $\mathbf{P}'$ will be accepted if and only if the product of all the $\tilde{\alpha}$ and $\tilde{\beta}$ values of the surrounding elements is increased and usually three cycles of quality smoothing will be sufficient.

Note that for both the length and element quality smoothing procedures, after the point $\mathbf{P}'$ is accepted, the metric tensor at $\mathbf{P}'$ will be retrieved from the background mesh.

Quadrilateral mesh conversion

In case that a pure quadrilateral mesh is required, a simple merging and subdivision procedure[26,27] will be used to convert the triangular mesh generated to a pure quadrilateral mesh. The conversion procedure consists of five essential steps.

(i) Merging of triangular elements

The merging algorithm used here is exactly the one previous developed in reference [28]. Diagonals of the meshes will be removed one by one (Fig. 8) according to the $\tilde{\beta}$ value of the resulting quadrilateral. The merging process will be stopped until no more triangles is left or when the further removal of diagonal will result in an element with a $\tilde{\beta}$ value less than 0.1. This in general will produce a mixed mesh consists of both triangles and quadrilaterals.

(ii) Mixed mesh quality enhancement

The quality of the mixed mesh will be enhanced by applying the length and element smoothing procedures described in the last section. Furthermore, a triangular element elimination process as shown in Fig. 9 will be carried out. In Fig. 9, after the triangles IT1 and IT2 are removed, the new node **Q** which is the midpoint of the nodes $\mathbf{P}_1$ and $\mathbf{P}_2$ will be formed.

(iii) Removal of isolated triangles

In order to further reduce the number of isolated triangles in the mixed mesh. All triangles in the mesh will be considered one by one. Quadrilateral elements surrounding the triangle (at most three) will be identified (Fig. 10). A pentagon will be created by merging the triangle with one of the quadrilaterals such that the angle deviation of the resulting pentagon, ϕ, defined as

$$\phi = \left|\theta_1 - \frac{\pi}{2}\right| + \left|\theta_2 - \frac{\pi}{2}\right| + \left|\theta_3 - \frac{\pi}{2}\right| \tag{28}$$

is minimized.

(iv) Division of elements

A pure quadrilateral mesh will be formed by dividing the triangles, quadrilaterals and pentagons in the mixed mesh as shown in Fig. 11. Since the subdivision process will reduce the elements in the mesh to approximately one quarter of their original sizes, whenever a quadrilateral mesh is required the initial metric tensor will be automatically multiplied by four. (Since the eigenvalues of the metric tensor is inversely proportional to the square of the principal element sizes.) Note that during the subdivision process, the mid-side node **C** as shown in Fig. 11 will be inserted in such a way that

$$\tilde{l}(\mathbf{M}, \mathbf{AC}) = \tilde{l}(\mathbf{M}, \mathbf{CB}) = 0.5 \tag{29}$$

while the interior node **D** will be placed at the centroid of the surrounding mid-side nodes.

(v) Quadrilateral mesh quality enhancement

After the quadrilateral mesh is formed, length and element quality smoothing procedures will be applied to improve the quality of the mesh.

4. Mesh generation examples

In this section, four mesh generation examples will be presented to demonstrate the performance of the new metric mesh generator. The element characteristics of the meshes are directed by some fictitious metric tensors. Both isotropic ($h_1=h_2$) and anisotropic ($h_1\neq h_2$) cases will be considered. In all the examples given in this section, the mesh generation procedure was started by defining the specified metric tensor *implicitly* over the nodal points of some very coarse initial meshes, which is then fed into the generator to imitate the situation during an adaptive analysis. For an easier interpretation of the metric tensors used, they are expressed in terms of the principal element sizes $\{h_1, h_2\}$ and the principal stretching directions $\{\mathbf{e}_1, \mathbf{e}_2\}$.

In all the examples except Example 3, the origin (0,0) of the co-ordinate axes is located at the lower left hand corner of the mesh. Five refinements were carried out to generate the final meshes.

Example 1: A simple metric intersection example

In this example, the effect of metric tensor intersection will be demonstrated. The problem domain under consideration is a square with edge length equal to 10 units. The initial mesh used is shown in Fig. 12a. The first metric tensor is

$$\mathbf{M}_1(x,y):\begin{cases} h_1 = 0.005 + 0.15r(x,y),\ h_2 = 15h_1 e^{-5.5r(x,y)} \\ \mathbf{e}_1 = \begin{pmatrix} -1 \\ 1 \end{pmatrix},\quad \mathbf{e}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \end{cases} \tag{30}$$

where r is the shortest distance between the point (x,y) and the line x=y. One can expect, as shown in Fig. 12b, the final triangular mesh generated is highly stretched along the line x=y.

The second metric used are defined by replacing r(x,y) in Eqn. 30 by the shortest distance from the line x+y=10 and reversing the vectors $\mathbf{e}_1$ and $\mathbf{e}_2$. i.e. $\mathbf{e}_1=(1,1)^T$ and $\mathbf{e}_2=(-1,1)^T$. The final triangular mesh generated is shown in Fig. 12c which is nearly a mirror image of Fig. 12b. The final triangular and quadrilateral meshes obtained when these two metrics are intersected (Eqn. 8) are shown in Figs. 12d and 12e respectively. Note that the stretching effects of the two metric tensors canceled each another near the centre of the square.

Example 2: A square plate with boundary layers

In this example, the square domain and the initial mesh used in Example 1 was reused. However, the metric tensors were changed such that the resulting meshes can be used in the adaptive analysis of shear boundary layers in plate bending problems[5]. Both isotropic and

anisotropic cases were considered and two metric tensors were used in both cases. For the isotropic case, the metric tensors used are

$$\begin{aligned} \mathbf{M}_1: &\quad h_1 = 0.05 + 0.2y^{\frac{5}{4}},\ h_2 = h_1 \\ \mathbf{M}_2: &\quad h_1 = 0.05 + 0.2x^{\frac{5}{4}},\ h_2 = h_1 \end{aligned} \tag{31}$$

and the principal stretching directions are immaterial. The final triangular and quadrilateral meshes generated are shown in Fig. 13a and 13b respectively. From these figures, it can be seen that small elements are concentrated along supported (the bottom and left hand side) edges of the plate where the boundary layer effect is strong[5]. However, as the shear forces of the plate will only vary rapidly in the *normal* direction of the supported edges, the computational cost needed during the finite element analysis can be much reduced by stretching the elements in the *tangential* direction of the supported edges. Hence, in the anisotropic case a stretching factor equal to 8.0 in the tangential direction was applied along the supported edges of the plate. The corresponding metric tensors are

$$\begin{aligned} \mathbf{M}_1': &\quad h_2 = 0.05 + 0.2y^{\frac{5}{4}},\ h_1 = 8h_2e^{-0.2079y} \\ \mathbf{M}_2': &\quad h_1 = 0.05 + 0.2x^{\frac{5}{4}},\ h_2 = 8h_1e^{-0.2079x} \end{aligned} \tag{32}$$

and for both $\mathbf{M}_1'$ and $\mathbf{M}_2'$, $\mathbf{e}_1=(1,0)^T$ and $\mathbf{e}_1=(0,1)^T$. Note that the stretching factor was selected in such a way that the meshes produced will become isotropic at the point (10,10) (upper right corner). The final meshes generated are shown in Figs. 13c and 13d and the stretching effects along the supported edges can be clearly seen. In fact, when comparing with the isotropic case, the number of nodes of the stretched meshes are only 17-19 percents of that of the corresponding isotopic meshes (Please refer to Table 1 for more details). Hence, in the adaptive analysis, the used of stretched meshes will result in a significant saving in computational resource.

Example 3. Meshing of a dam

In this example, a dam was considered and the initial mesh used is shown in Fig. 14a. The origin of this problem is the point **A** shown in Fig. 14a. The height of the dam is equal to 70 units so that the co-ordinates of the point **B** are (-10,70). For this example, in the isotropic case, small size elements were generated around an ellipse with centre at the point (10,0). The x and y axes are the major and minor axes of the ellipse respectively. The principal radii in

the x and y directions are equal to 50 and 10 units respectively. For the isotropic case, the metric tensor used is

$$\mathbf{M}\text{: } h_1 = 0.005 + 3r^{1.25},\ h_2 = h_1 \tag{33a}$$

where r is the normalized distance from the ellipse, defined as

$$r = \sqrt{\left|\left(\frac{x-10}{50}\right)^2 + \left(\frac{y}{25}\right)^2 - 1\right|} \tag{33b}$$

The final meshes generated are shown in Figs. 14b and 14c.

For the anisotropic case, the metric tensor was designed in such a way that elements were elongated in the tangential direction of the ellipse and the stretching effect dies down exponentially as the distance from the ellipse increases. The anisotropic metric map, $\mathbf{M}'$ used is

$$\mathbf{M}'\text{: } \begin{cases} h_2 = 0.005 + 3r^{1.25},\ h_1 = \chi h_2 \\ \mathbf{e}_1 = \begin{pmatrix} t_1 \\ t_2 \end{pmatrix},\ \mathbf{e}_2 = \begin{pmatrix} -t_2 \\ t_1 \end{pmatrix} \end{cases} \tag{34a}$$

where

$$\chi = \max(1,\ 8e^{-0.693r}),\quad t_1 = \frac{4y}{\sqrt{(4y)^2 + (x-10)^2}},\quad t_2 = \frac{-(x-10)}{\sqrt{(4y)^2 + (x-10)^2}} \tag{34b}$$

Figs. 14d and 14e show the final anisotropic meshes generated and one can see that the stretching effect is indeed achieved.

Example 4: Domain with internal openings

In the last example, a problem domain with three internal openings was considered. The width of the problem domain is equal to 240 units while the height is 160 units. The initial mesh is shown in Fig. 15a and in the isotropic case, the sizes of elements in the mesh were made proportional to r, the shortest distance between the point under consideration and the internal boundary. The metric **M** used can be expressed as

$$\mathbf{M}\text{: } h_1 = \min(25,\ 0.8 + 0.04r^{1.5}),\ \ h_2 = h_1 \tag{35}$$

and the meshes generated are shown in Fig. 15b and 15c.

For the anisotropic case, at any point in the problem domain, the elements were stretched in the tangential direction of nearest internal boundary segment. The stretching factor applied is

proportional to the radius of curvature at that boundary segment. The anisotropic metric tensor used can be written as

$$\mathbf{M}'(x,y) = \left\{ \begin{array}{c} h_2 = \min(25,\ 0.8 + 0.04r^{1.5}),\ h_1 = \min(25,\ \chi h_2) \\ \mathbf{e}_1 = \begin{pmatrix} t_1 \\ t_2 \end{pmatrix},\quad \mathbf{e}_1 = \begin{pmatrix} -t_2 \\ t_1 \end{pmatrix} \end{array} \right\} \tag{36a}$$

where the stretching factor χ is given by

$$\chi = \max(1,\ \frac{10}{(r+1)^{1/4}} e^{-c}) \tag{36b}$$

and

$$\begin{pmatrix} t_1 \\ t_2 \end{pmatrix} = \frac{\mathbf{P_B} - \mathbf{P_A}}{|\mathbf{AB}|} \tag{36c}$$

In Eqns. 36b and 3bc, the segment **AB** is the nearest internal boundary segment from the point (x,y). $\mathbf{P_A}$ and $\mathbf{P_B}$ are the co-ordinates of the points **A** and **B** respectively and *c* is the curvature of the domain boundary at the segment **AB**. Hence, one can expected that stretched elements will be generated along the internal boundary at where the curvature is small as shown in Figs. 15d and 15e.

Shape quality and grading of meshes generated

A summary of the mesh characteristics for the final meshes generated shown in Figs. 12 to 15 is given in Tables 1 to 4. From these tables, it can be concluded that the quality of all the triangular meshes generated are good and the geometrical mean $\tilde{\alpha}$ values obtained are all greater than 0.85 while the minimum values are always greater than 0.4. For the quadrilateral meshes generated, as the $\tilde{\beta}$ quality is more sensitive than the $\tilde{\alpha}$ quality to element shape distortion, lower values are obtained in all the quadrilateral meshes when comparing with the corresponding triangular meshes. However, in all the cases, the mean $\tilde{\beta}$ values are all greater than 0.6 while the minimum values are greater than 0.18. Hence, we can conclude that the quality of the meshes generated as better than acceptable.

From Tables 1 to 4, it can be seen that for all triangular meshes, more than 96% of all the edges generated with lengths within the range [0.5,1.5]. In fact, detail statistics reveal that more than 90% of the edges generated have lengths within the range [0.7,1.3]. For quadrilateral meshes, as triangular meshes with double element size are first generated and then subdivided, the final grading of the quadrilateral meshes is in general lower than the

corresponding triangular meshes (last three columns of Tables 1-4). However, in all the examples, more than 80% of the edges generated have lengths within the range [0.5,1.5]. Furthermore, it can be seen that for all the final quadrilateral meshes generated, the numbers of elements are approximately equal to one half of the corresponding triangular meshes while the numbers of nodes are approximately the same. The results shown in Figs. 12 to 15 also confirmed that the mesh patterns of the triangular meshes are preserved in all the quadrilateral meshes.

For the convergence rate of the mesh generation process, the convergence history of the edge length distributions for Example 4, anisotropic case, are shown in Figs. 16 and 17 for triangular and quadrilateral meshes respectively. (Results from other examples are very similar and hence omitted here). From these figures, it can be seen that the edge length distributions for both types of mesh converged rapidly and attained their final forms within only two to three iterations. Hence, the mesh generation algorithm can be used in an adaptive analysis for the generation of well graded finite element meshes.

5. Conclusions and future investigations

In this paper, the advancing front technique has been combined with the metric specification approach for the generation of adaptive finite element meshes with element characteristics compatible with the input requirements. A novel triangular element formation algorithm based on the concept of unit vector rotation in the normalized space is introduced and it is found that this algorithm can consistently lead to satisfactory element grading and stretching effects. New mesh quality enhancement procedures have also been presented for improving the shape quality and grading of the meshes generated. A simple conversion scheme based on the method of subdivision and merging is employed to generate complete quadrilateral meshes. Several mesh generation examples involving rapid grading of element size and stretching ratio have been used to test the performance and robustness of the mesh generator. Satisfactory results have been obtained in all cases.

For the speed of the new mesh generation scheme, since all geometrical measurements are computed using the metric specification, the new scheme will incur a greater computational cost when compared with other traditional advancing front mesh generators. However, it is found that the *operation complexity* of the new mesh generation scheme is the same as the traditional advancing front mesh generator developed earlier[26] and is equal to $O(NE \cdot \log(NE))$ for a mesh with NE elements. The generation speed of the current algorithm

is about 100 elements per second on a low end 100MHz Pentium PC and it is fast enough of most practical applications.

One possible extension of the current work includes the use of the systemic merging technique[29] for the complete merging of triangular elements during the conversion of pure quadrilateral mesh. As in such a case it is not necessary to first double the mesh density during the triangular mesh generation process, a further increase in the grading quality of the final mesh is expected. Other potential areas of further research will be the use of the metric specification approach in surface and 3D advancing front mesh generations and research on these areas are currently under serious consideration.

6. Appendices

Appendix A: Metric tensors for 3D and surface mesh generations

For 3D mesh generation, the metric tensor, $\mathbf{M}_{3D}$, is simply a three by three positive definite matrix defined as

$$\mathbf{M}_{3D} = \begin{bmatrix} a & b & d \\ b & c & e \\ d & e & f \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix}^T \qquad \text{(A1)}$$

such that $\mathrm{Det}(\mathbf{M}_{3D}) > 0$ and $\lambda_i = \dfrac{1}{(h_i)^2} > 0$ for $i = 1,2,3$. The length scale transformation for an elementary vector $\mathbf{dx}$ can then be expressed as

$$\tilde{l}^2 = \mathbf{d\xi}^T \mathbf{d\xi} = \left(\mathbf{M}_{3D}^{1/2} \mathbf{dx}\right)^T \mathbf{M}_{3D}^{1/2} \mathbf{dx} = \mathbf{dx}^T \mathbf{M}_{3D} \mathbf{dx} \qquad \text{(A2)}$$

In such a case the matrix $\mathbf{M}_{3D}^{1/2}$ and its inverse will be most conveniently expressed in terms of the eigenpairs of $\mathbf{M}_{3D}$. That is

$$\mathbf{M}_{3D}^{1/2} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix} \begin{bmatrix} \lambda_1^{1/2} & 0 & 0 \\ 0 & \lambda_2^{1/2} & 0 \\ 0 & 0 & \lambda_3^{1/2} \end{bmatrix} \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix}^T \qquad \text{(A3a)}$$

$$\mathbf{M}_{3D}^{-1/2} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix} \begin{bmatrix} \lambda_1^{-1/2} & 0 & 0 \\ 0 & \lambda_2^{-1/2} & 0 \\ 0 & 0 & \lambda_3^{-1/2} \end{bmatrix} \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix}^T \qquad \text{(A3b)}$$

For surface mesh generation, the co-ordinates of a point $\mathbf{P}=(x,y,z)^T$ on the surface can usually be expressed as a bivariate mapping such that

$$(x, y, z)^T = \mathbf{r}(u, v) \qquad \text{(A4)}$$

Hence, the transformation of an elementary vector can be expressed as

$$\mathbf{dx} = (dx, dy, dz)^T = \begin{pmatrix} \dfrac{\partial x}{\partial u} & \dfrac{\partial x}{\partial v} \\ \dfrac{\partial y}{\partial u} & \dfrac{\partial y}{\partial v} \\ \dfrac{\partial z}{\partial u} & \dfrac{\partial z}{\partial v} \end{pmatrix} \begin{pmatrix} du \\ dv \end{pmatrix} = \begin{pmatrix} \mathbf{r}_{,u} & \mathbf{r}_{,v} \end{pmatrix} \mathbf{du} \qquad \text{(A5)}$$

The length scale transformation between the u-v and the x-y-z co-ordinate systems is thus given by

$$\left|\mathbf{dx}\right|^2 = \mathbf{dx}^{\mathsf{T}}\mathbf{dx} = \left[\left(\mathbf{r}_{,\mathbf{u}}\ \mathbf{r}_{,\mathbf{v}}\right)\mathbf{du}\right]^{\mathsf{T}}\left(\mathbf{r}_{,\mathbf{u}}\ \mathbf{r}_{,\mathbf{v}}\right)\mathbf{du} = \mathbf{du}^{\mathsf{T}}\begin{pmatrix}\mathbf{r}_{,\mathbf{u}}\,\mathbf{r}_{,\mathbf{u}} & \mathbf{r}_{,\mathbf{u}}\,\mathbf{r}_{,\mathbf{v}}\\ \mathbf{r}_{,\mathbf{v}}\,\mathbf{r}_{,\mathbf{u}} & \mathbf{r}_{,\mathbf{v}}\,\mathbf{r}_{,\mathbf{v}}\end{pmatrix}\mathbf{du} \qquad \text{(A6)}$$

Comparing Eqn. A6 with Eqn. 10, the metric tensor corresponding to the surface transformation, $\mathbf{M}_{uv}$, is therefore given by

$$\mathbf{M}_{uv} = \begin{pmatrix}\mathbf{r}_{,\mathbf{u}}\,\mathbf{r}_{,\mathbf{v}} & \mathbf{r}_{,\mathbf{u}}\,\mathbf{r}_{,\mathbf{v}}\\ \mathbf{r}_{,\mathbf{v}}\,\mathbf{r}_{,\mathbf{u}} & \mathbf{r}_{,\mathbf{v}}\,\mathbf{r}_{,\mathbf{v}}\end{pmatrix} \qquad \text{(A7)}$$

Usually, during surface mesh generation, the element characteristics are most conveniently defined with respect to the 3D co-ordinate system using Eqn. A2. However, mesh generation is usually carried out in the u-v space to reduce the computational cost. Thus, a single metric tensor is required to describe the combined effect of the user specifications and the surface mapping. This can be obtained by combing Eqn. A2 with Eqn. A5.

$$\begin{aligned}\tilde{l}^2 &= \mathbf{dx}^{\mathsf{T}}\mathbf{M}_{3D}\mathbf{dx}\\ &= \left[\left(\mathbf{r}_{,\mathbf{u}}\quad \mathbf{r}_{,\mathbf{v}}\right)\mathbf{du}\right]^{\mathsf{T}}\mathbf{M}_{3D}\left(\mathbf{r}_{,\mathbf{u}}\quad \mathbf{r}_{,\mathbf{v}}\right)\mathbf{du}\\ &= \mathbf{du}^{\mathsf{T}}\left[\left(\mathbf{r}_{,\mathbf{u}}\quad \mathbf{r}_{,\mathbf{v}}\right)^{\mathsf{T}}\mathbf{M}_{3D}\left(\mathbf{r}_{,\mathbf{u}}\quad \mathbf{r}_{,\mathbf{v}}\right)\right]\mathbf{du}\end{aligned} \qquad \text{(A8)}$$

Hence, the 2×2 matrix, $\mathbf{M}_{sur}$, defined as

$$\mathbf{M}_{sur} = \left(\mathbf{r}_{,\mathbf{u}}\quad \mathbf{r}_{,\mathbf{v}}\right)^{\mathsf{T}}\mathbf{M}_{3D}\left(\mathbf{r}_{,\mathbf{u}}\quad \mathbf{r}_{,\mathbf{v}}\right) \qquad \text{(A9)}$$

is the metric tensor which transforms the elementary vector **du** to the normalized space.

Appendix B: Segment length evaluation

Let **A** and **B** be the end points of a segment with metric tensors $\mathbf{M_A}$ and $\mathbf{M_B}$ respectively. One way to compute the length $\tilde{l}(\mathbf{M}, \mathbf{AB})$ is by applying numerical integration to Eqn. 1. However, when the metric tensor varies rapidly along **AB**, Eqn. 1 may not be evaluated accurately even when a high order integration rule is used. An alternative method is to use the subdivision procedure described below.

(i) Divide the segment **AB** into two sub-segments **AC** and **CB** where **C** is the midpoint of **AB**. The current number of sub-segment, NS, is set to 2.

(ii) Compute the lengths of all the current sub-segments, $\tilde{l}_i$, i=1,..NS, by using a low order integration rule.

(iii) Let $\tilde{l}_{max}$ be the maximum length of all current sub-segments. That is

$$\tilde{l}_{max} = \max_{i=1,\ldots,NS} (\tilde{l}_i) \tag{A10}$$

If $\tilde{l}_{max}$ is less than a given tolerance ε, then the length of the segment **AB** is computed as

$$\tilde{l}(\mathbf{M}, \mathbf{AB}) = \sum_{i=1}^{NS} \tilde{l}_i \tag{A11}$$

(iv) Otherwise, all sub-segments are further divided at their midpoints and the value of NS is doubled and goto step (ii).

In the current implementation, the length of a given sub-segment **PQ**, $\tilde{l}(\mathbf{PQ})$, is calculated by the follow two-point formula.

$$\tilde{l}(\mathbf{PQ}) = \frac{1}{2}\left(\sqrt{\mathbf{PQ}^T \mathbf{M_P} \mathbf{PQ}} + \sqrt{\mathbf{PQ}^T \mathbf{M_Q} \mathbf{PQ}}\right) \tag{A12}$$

where $\mathbf{M_P}$ and $\mathbf{M_Q}$ are the metric tensors at **P** and **Q** respectively. The tolerance used in step (iii) is set as 0.25. This subdivision method usually terminates within three iterations.

Note that this subdivision procedure can also be used for locating the midpoint of the segment with respect to the metric tensor. The only additional effort required is to find out the jth sub-segment such that

$$\sum_{i=1}^{j-1} \tilde{l}_i \le 0.5 \le \sum_{i=1}^{j} \tilde{l}_i \tag{A13}$$

Then the position of the midpoint can be obtained by linear interpolation.

Appendix C: Co-ordinates for the points **C2** *and* **C3**

The co-ordinates of the point **C2**=$(u_{C2}, v_{C2})^T$ are

$$\begin{pmatrix} u_{C2} \\ v_{C2} \end{pmatrix} = \begin{pmatrix} u_B \\ v_B \end{pmatrix} + \frac{\sqrt{3}}{2d \cdot \tilde{l}(\mathbf{M_B}, \mathbf{AB})} \begin{pmatrix} \frac{d}{\sqrt{3}} + b & c \\ -a & \frac{d}{\sqrt{3}} - b \end{pmatrix} \begin{pmatrix} u_A - u_B \\ v_A - v_B \end{pmatrix} \qquad \text{(A14)}$$

where $\mathbf{M_B} = \begin{pmatrix} a & b \\ b & c \end{pmatrix}$ and $d = \sqrt{ac - b^2}$.

Similarly, the co-ordinates of the point **C3**=$(u_{C3}, v_{C3})^T$ are

$$\begin{pmatrix} u_{C3} \\ v_{C3} \end{pmatrix} = \begin{pmatrix} u_D \\ v_D \end{pmatrix} + \frac{\sqrt{3}}{2d \cdot \tilde{l}(\mathbf{M_D}, \mathbf{DB})} \begin{pmatrix} -b & -c \\ a & b \end{pmatrix} \begin{pmatrix} u_B - u_D \\ v_B - v_D \end{pmatrix} \qquad \text{(A15)}$$

where a,b,c and d are, of course, terms corresponding to the metric tensor $\mathbf{M_D}$.

7. References

1. J. L. Graysmith and C. T. Shaw, Automated procedures for Boolean operations on finite element meshes, *Eng. Comput.* **14,** 702-717 (1997)

2. B. Boroomand and O. C. Zienkiewicz, An improved REP recovery and the effectivity robustness test, *Int. J. Numer. Methods Engrg.,* **40,** 3247-3277 (1997)

3. O. C. Zienkiewicz and J. Z. Zhu, Adaptivity and mesh generation, *Int. J. Numer. Methods Engrg.,* **32,** 783-810 (1991)

4. T. Lee, H. C. Park and S. W. Lee, A superconvergent stress recovery technique with equilibrium constraint, *Int. J. Numer. Methods Engrg.,* **40** 1139-1160 (1997)

5 C. K. Lee and R. E. Hobbs, Automatic adaptive refinement for plate bending problems using Reissner-Mindlin plate bending elements, *Int. J. Numer. Methods Engrg.,* **41** 1-63 (1998)

6. P. L. George, Automatic mesh generation, application to finite element methods, John Wiley & Sons Publishing Company, 1991.

7. K. Ho-Le, Finite element mesh generation methods: A review and classification, *Comput. Aided Des.,* **20,** 27-38 (1988)

8. P. L. Baehmann, S. L. Wittchen, M. S. Shephard, K. R. Grice and M. A. Yerry, Robust, Geometrically based, automatic two-dimensional mesh generation, *Int. J. Numer. Methods Engrg.,* **24,** 1043-1078 (1987)

9. M. S. Shephard and M. K. Georges, Automatic three-dimensional mesh generation by the finite octree technique, *Int. J. Numer. Methods Engrg.,* **32,** 709-749 (1991)

10. D. F. Watson, Computing the n-dimensional Delaunary tessellation with application to Voronoi polytopes, *Comput. J.,* **24,** 167-172 (1981)

11. H. Borouchaki and P. J. Frey, Adaptive triangular-quadrilateral mesh generation, *Int. J. Numer. Methods Engrg.,* **41,** 915-934 (1998)

12. P. L. George and E. Seveno, The advancing-front mesh generation method revisited, *Int. J. Numer. Methods Engrg.,* **37,** 3605-3619 (1994)

13. S. H. Lo, Automatic mesh generation and adaptation by using contours, Int. *J. Numer. Methods Engrg.,* **31,** 689-707 (1991)

14. J. Peraire, J. Peiro, K. Morgan and O. C. Zienkiewicz, Finite element mesh generation and adaptive procedure for CFD, *Proceedings of GAMNI/SMAI Conference on Automatic and adaptive mesh generation,* Grenoble, France, October 1-2, 1987.

15. E. Hinton and H. C. Huang, Shear forces and twisting moments in plates using Mindlin elements, *Engrg. Computations.,* **3,** 129-142 (1986)

16. W. J. Gordon and C. A. Hall, Construction of curvilinear co-ordinate systems and applications to mesh generation, *Int. J. Numer. Methods Engrg.,* **7,** 461-477 (1973)

17. S. H. Wu and J. F. Abel, Representation and discretization of arbitrary surfaces for finite element shell analysis, *Int. J. Numer. Methods Engrg.,* **14,** 813-836 (1979)

18. W. H. Frey, Selective refinement: A new strategy for automatic node placement in graded triangular meshes, *Int. J. Numer. Methods Engrg.,* **24,** 2183-2200 (1987)

19. R. Lohner, Automatic unstructured grid generators, *Fin. Ele. Anal. Des.,* **25,** 111-134 (1997)

20. P. Moller and P. Hansbo, On advancing front mesh generation in three dimensions, *Int. J. Numer. Methods Engrg.,* **38,** 3551-3569 (1995)

21. H. Borouchaki, P. L. George, F. Hecht, P. Laug and E. Saltel, Delaunay mesh generation governed by metric specifications. Part I. Algorithms, *Fin. Ele. Anal. Des.,* **25,** 61-83 (1997)

22. H. Borouchaki, P. L. George, F. Hecht, P. Laug and E. Saltel, Delaunay mesh generation governed by metric specifications. Part II. Applications, *Fin. Ele. Anal. Des.,* **25,** 85-109 (1997)

23. G. F. Simmons, Introduction to topology and modern analysis, McGraw-Hill International Editions, 1963.

24. Bonet J, Peraire J. An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems, *Int. J. Numer. Methods Engrg.,* **31,** 1-17 (1991)

25. S. H. Lo, Generating quadrilateral elements on plane and over curved surfaces, *Comput. Struct.,* **31,** 421-426 (1989)

26. C. K. Lee and R. E. Hobbs, Automatic adaptive finite element mesh generation over rational B-spline surfaces, *Comput. and Struct.,* **69,** 577-608 (1998)

27. D. O. Potyondy, P. A. Wawrzynek and A. R. Ingraffea, An algorithm to generate quadrilateral or triangular element surface meshes in arbitrary domains with applications to crack propagation, *Int. J. Numer. Methods Engrg.,* **38,** 2677-2701 (1995)

28. S. H. Lo and C. K. Lee, On using meshes of mixed element types in adaptive finite element analysis, *Fin. Ele. Anal. and Des.,* **11,** 307-336 (1992)

29. C. K. Lee and S. H. Lo, A new scheme for the generation of a graded quadrilateral mesh, *Comput. and Struct.*, **52,** 847-857 (1994).

Mesh	Type	NN	NE	ND	$\tilde{\alpha}/\tilde{\beta}_{mean}$	$\tilde{\alpha}/\tilde{\beta}_{min}$	$ND_{0.5\text{-}1.5}(\%)$	$\tilde{l}_{min}$	$\tilde{l}_{max}$
Fig. 12b	A, T	423	748	1170	0.967	0.652	99.02	0.55	1.45
Fig. 12c	A, T	423	743	1170	0.967	0.651	99.81	0.55	1.45
Fig. 12d	A, T	1704	3266	4969	0.967	0.601	99.94	0.55	1.60
Fig. 12e	A, Q	1651	1582	3232	0.714	0.185	97.65	0.35	2.10

Table 1. Characteristics of meshes for Example 1

Mesh	Type	NN	NE	ND	$\tilde{\alpha}/\tilde{\beta}_{mean}$	$\tilde{\alpha}/\tilde{\beta}_{min}$	$ND_{0.5\text{-}1.5}(\%)$	$\tilde{l}_{min}$	$\tilde{l}_{max}$
Fig. 13b	I, T	2697	4958	7654	0.964	0.729	99.28	0.55	1.75
Fig. 13c	I, Q	2801	2584	5384	0.720	0.224	98.86	0.35	1.90
Fig. 13d	A, T	508	916	1423	0.954	0.536	99.84	0.65	1.75
Fig. 13e	A, Q	471	422	892	0.673	0.201	97.32	0.35	1.85

Table 2. Characteristics of meshes for Example 2

Mesh	Type	NN	NE	ND	$\tilde{\alpha}/\tilde{\beta}_{mean}$	$\tilde{\alpha}/\tilde{\beta}_{min}$	$ND_{0.5\text{-}1.5}(\%)$	$\tilde{l}_{min}$	$\tilde{l}_{max}$
Fig. 14b	I, T	5495	10871	16365	0.928	0.574	95.71	0.40	2.2
Fig. 14c	I, Q	4905	4843	9747	0.706	0.223	86.85	0.25	2.70
Fig. 14d	A, T	1292	2482	3773	0.857	0.433	95.47	0.50	2.10
Fig. 14e	A, Q	1077	1024	2102	0.602	0.189	82.45	0.20	2.95

Table 3. Characteristics of meshes for Example 3

Mesh	Type	NN	NE	ND	$\tilde{\alpha}/\tilde{\beta}_{mean}$	$\tilde{\alpha}/\tilde{\beta}_{min}$	$ND_{0.5\text{-}1.5}(\%)$	$\tilde{l}_{min}$	$\tilde{l}_{max}$
Fig. 15b	I, T	3362	6128	9492	0.958	0.510	99.48	0.50	1.75
Fig. 15c	I, Q	3478	3181	6661	0.674	0.201	97.77	0.25	2.20
Fig. 15d	A, T	1061	1964	3027	0.909	0.462	98.76	0.40	2.35
Fig. 15e	A, Q	1158	1080	2240	0.641	0.181	95.41	0.25	2.80

Table 4. Characteristics of meshes for Example 4

Legends for Tables 1-4

Type = Type of mesh, I = isotropic, A = anisotropic, T = triangular, Q = quadrilateral
NN = Number of nodes in the mesh
NE = Number of elements in the mesh
ND = Number of edges in the mesh
$\tilde{\alpha}/\tilde{\beta}_{mean}$ = Geometrical mean $\tilde{\alpha}$ ($\tilde{\beta}$) value of all triangles (quadrilaterals) in the mesh
$\tilde{\alpha}/\tilde{\beta}_{min}$ = Minimum $\tilde{\alpha}$ ($\tilde{\beta}$) value of all triangles (quadrilaterals) in the mesh
$ND_{0.5\text{-}1.5}$(%) = Percentage of edges with length (with respect to the metric tensor) inside the range [0.5,1.5]
$\tilde{l}_{min}$ = Minimum edge length (with respect to the metric tensor) of the mesh
$\tilde{l}_{max}$ = Maximum edge length (with respect to the metric tensor) of the mesh

Pre-printed version for the paper appeared in the journal
Engineering Computations, Vol. 16, 1999, 230-263

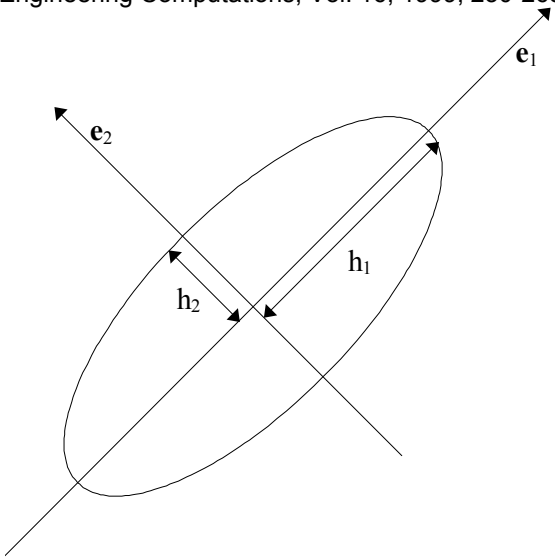


Figure 1. Ellipse defined by the metric tensor **M**

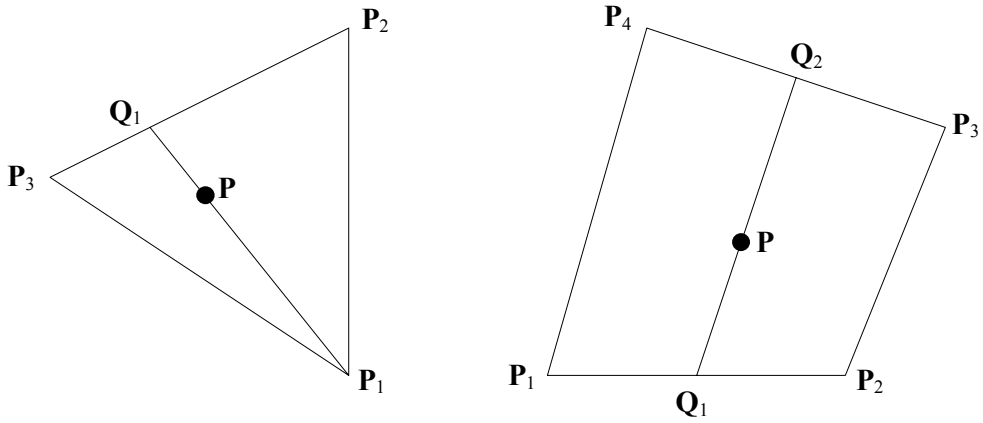


Figure 2. Interpolation of metric at point **P** inside a background element

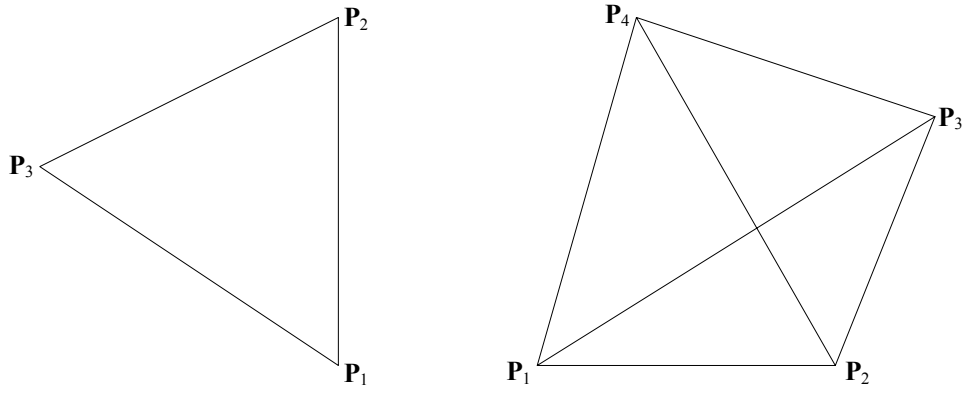


Figure 3. A triangular and a quadrilateral element

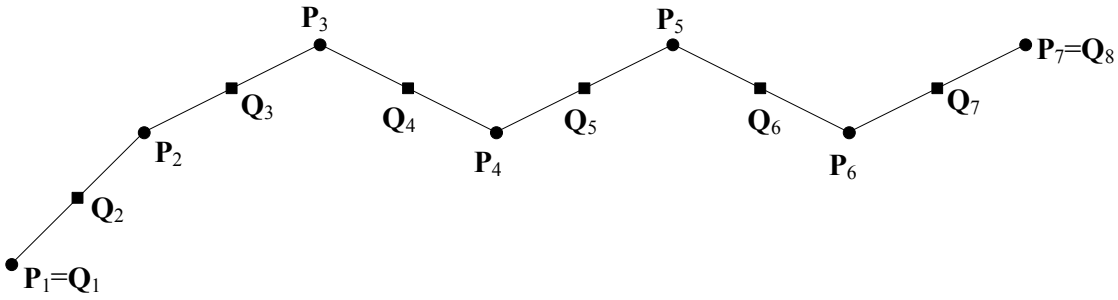


Figure 4. Generation of boundary segments

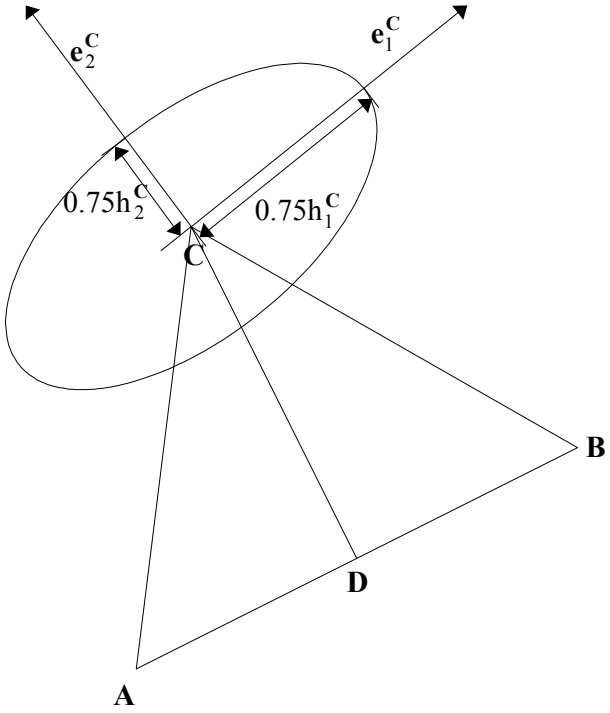


Figure 5. Formation of element

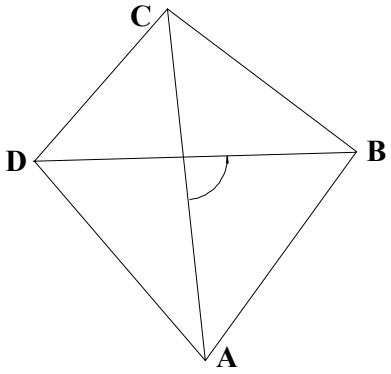


Figure 6. Swapping of diagonal

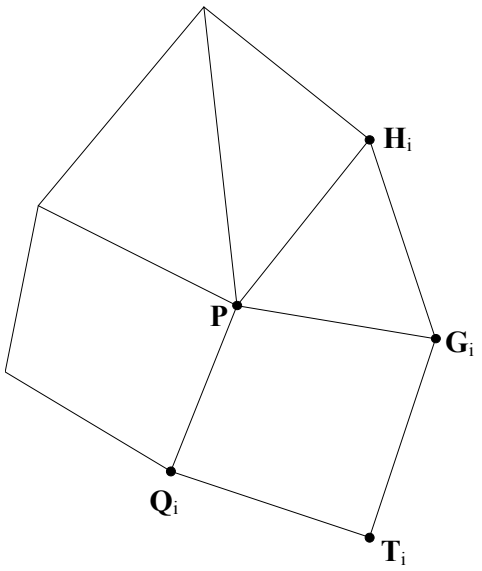


Figure 7. Smoothing of element

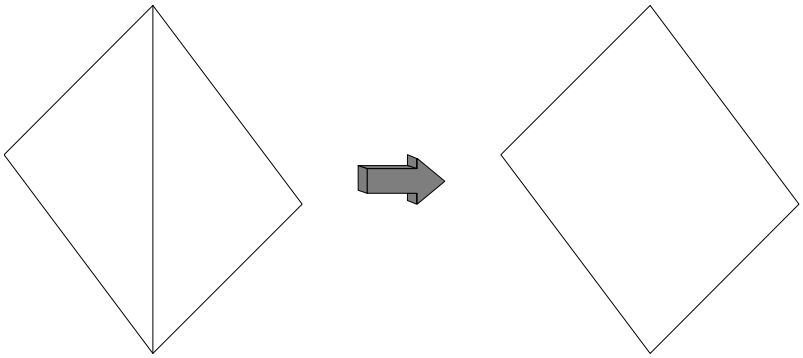


Figure 8. Merging of triangles

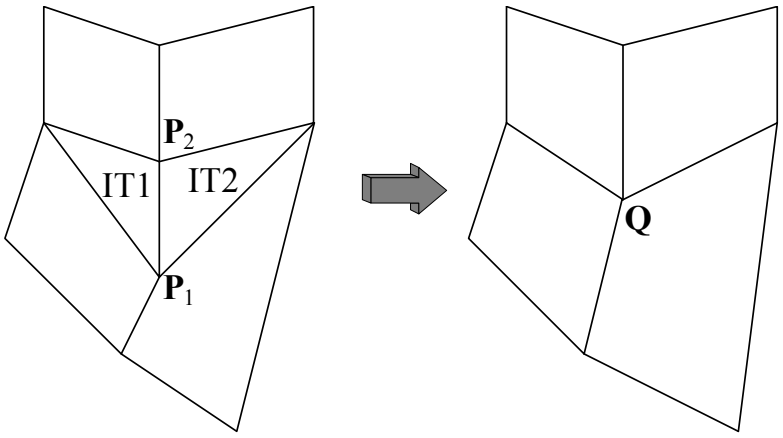


Figure 9. Triangular elements elimination

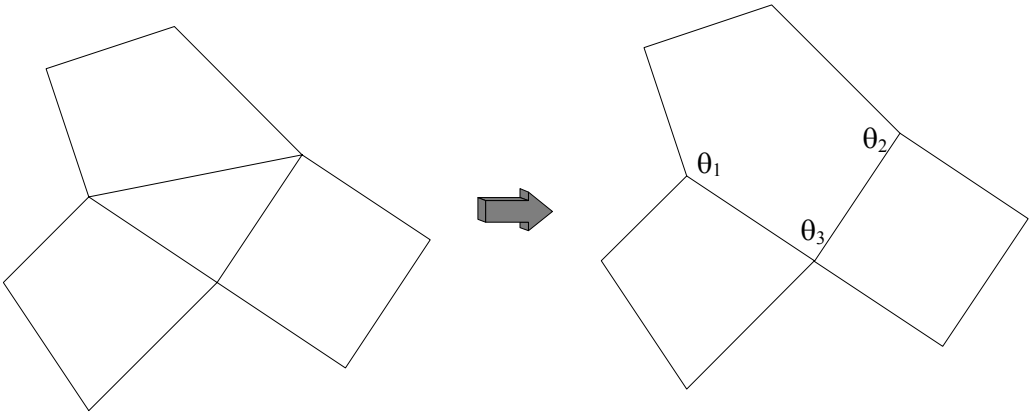


Figure 10. Formation of pentagon

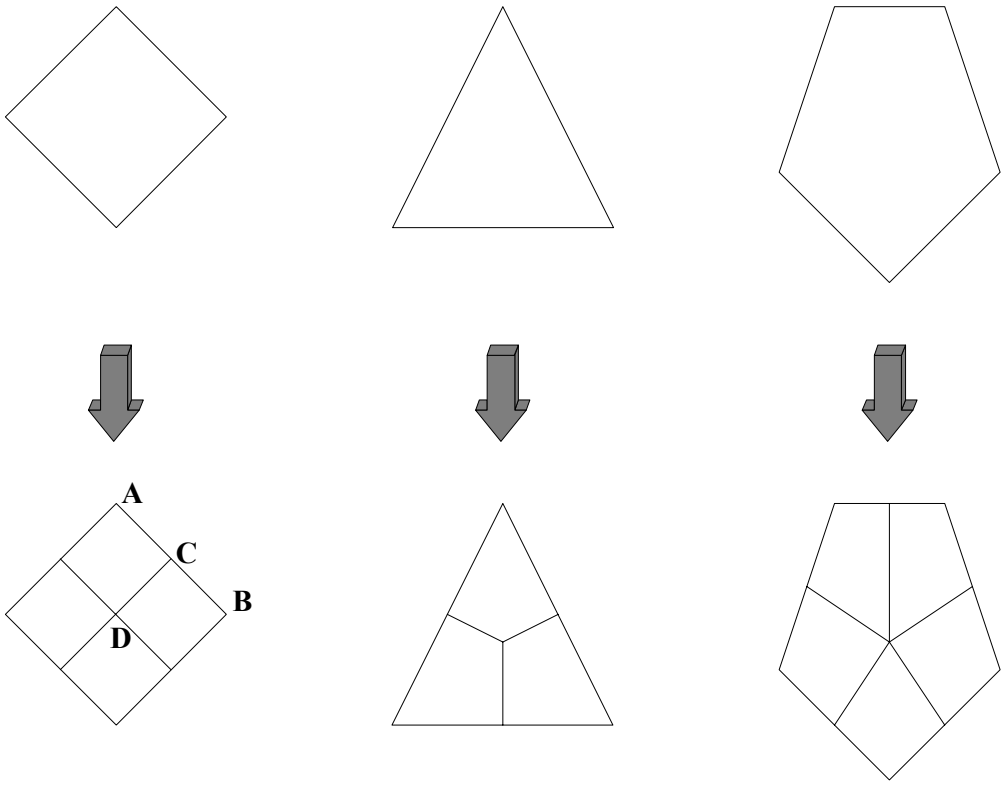
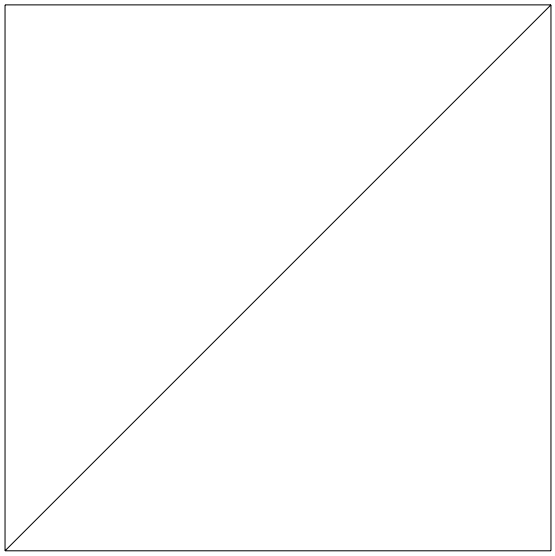
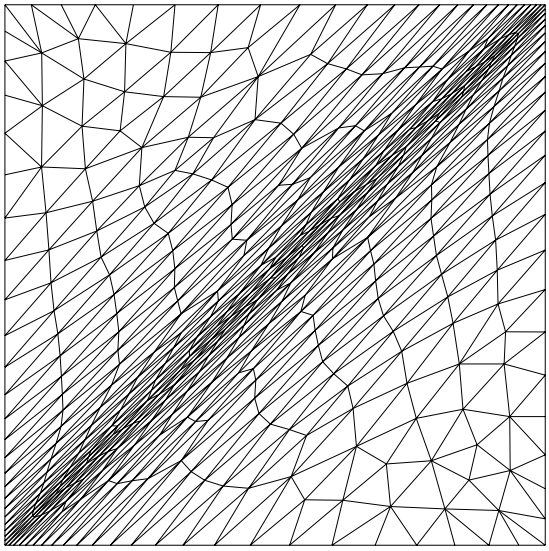


Figure 11. Divisions of elements

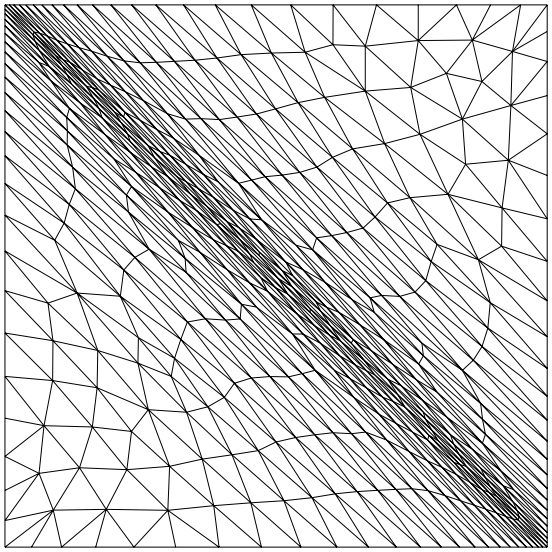


(a) Initial mesh for Examples 1 and 2

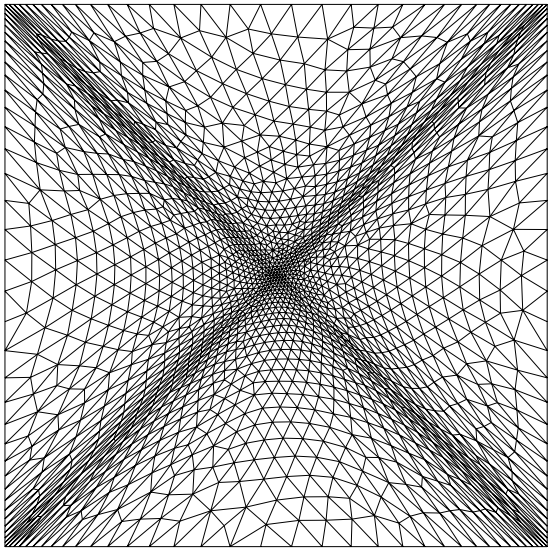


(b) Final triangular mesh, $\mathbf{M}_1$

Figure 12. Example 1

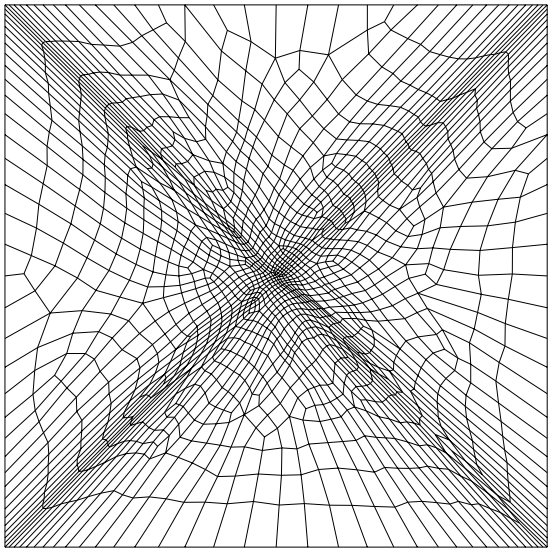


(c) Final triangular mesh, $\mathbf{M}_2$



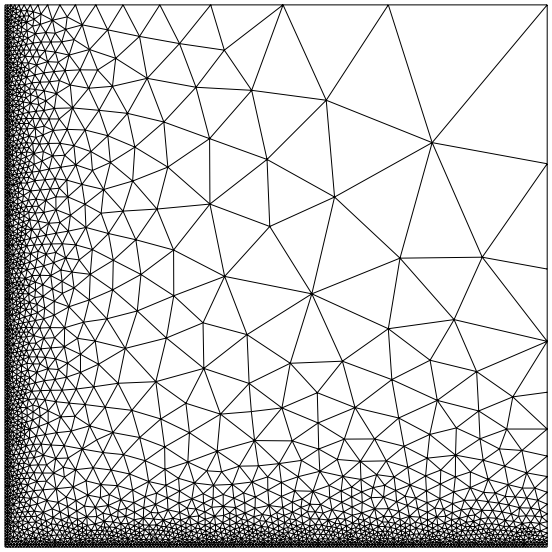
(d) Final triangular mesh, $\mathbf{M}_1 \cap \mathbf{M}_2$

Figure 12. Example 1 (Continued)

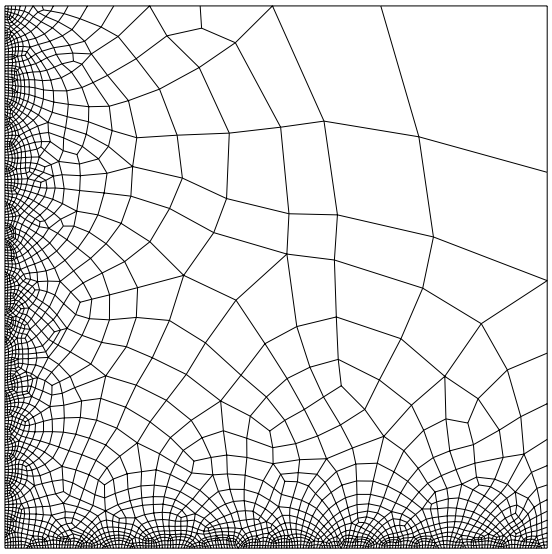


(e) Final quadrilateral mesh, $\mathbf{M}_1 \cap \mathbf{M}_2$

Figure 12. Example 1 (Continued)

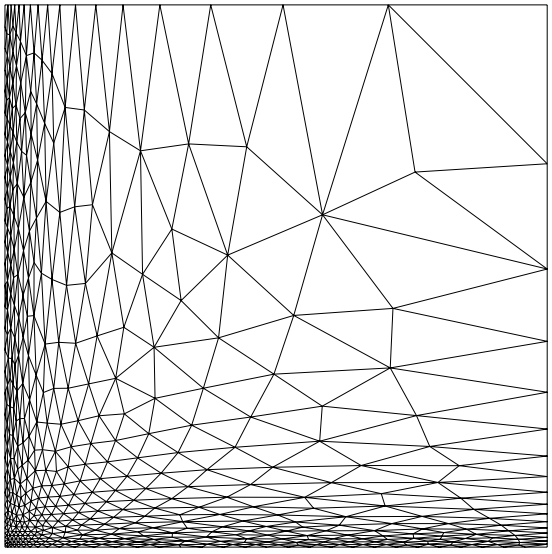


(a) Final triangular mesh, isotropic case

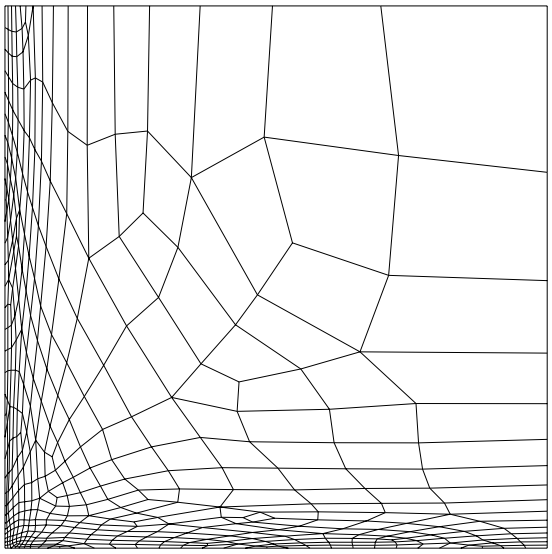


(b) Final quadrilateral mesh, isotropic case

Figure 13. Example 2

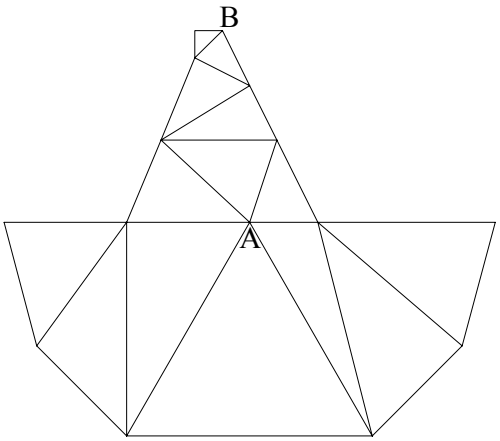


(c) Final triangular mesh, anisotropic case

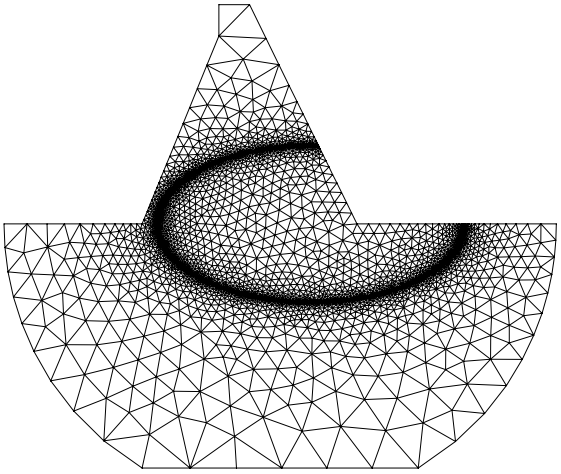


(d) Final quadrilateral mesh, anisotropic case

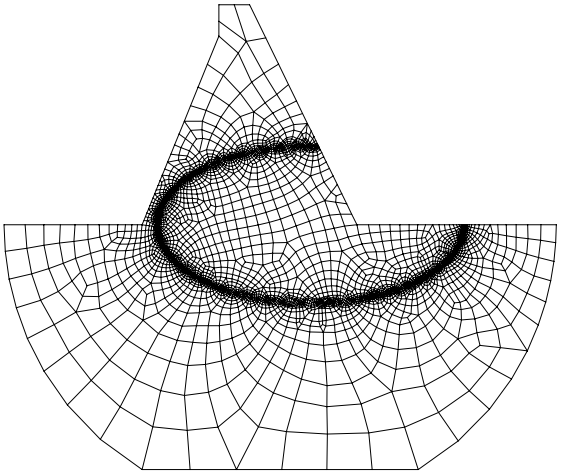
Figure 13. Example 2 (Continued)



(a) Initial mesh

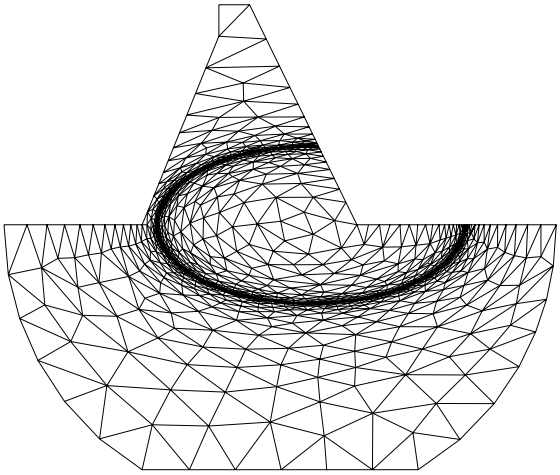


(b) Final triangular mesh, isotropic case

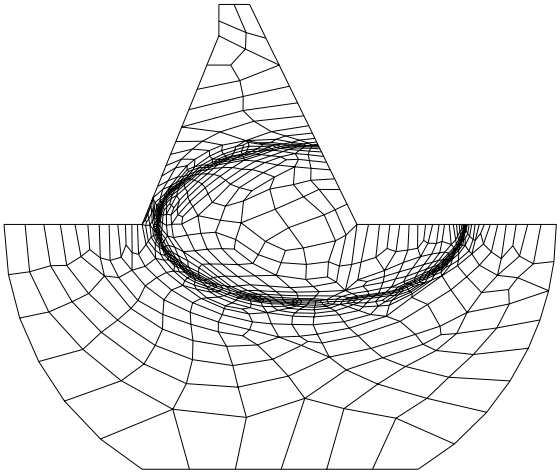


(c) Final quadrilateral mesh, isotropic case

Figure 14. Example 3

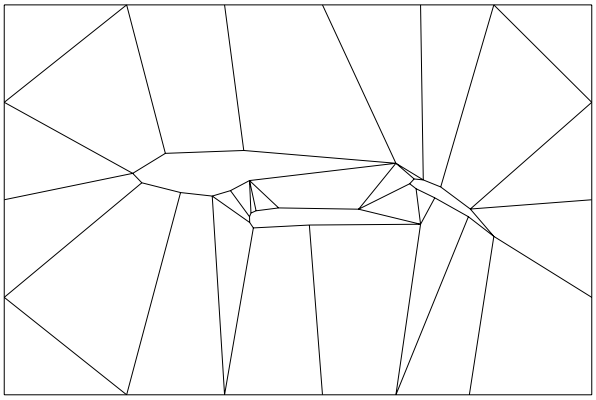


(d) Final triangular mesh, anisotropic case

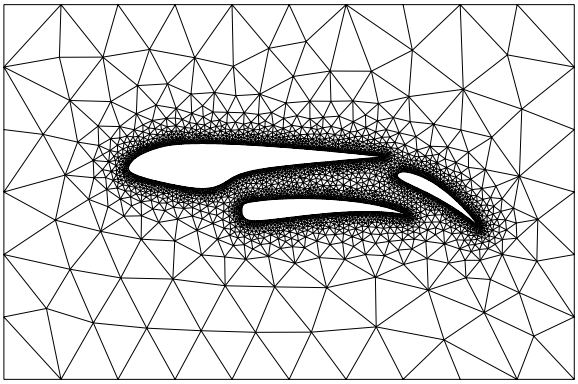


(e) Final quadrilateral mesh, anisotropic case

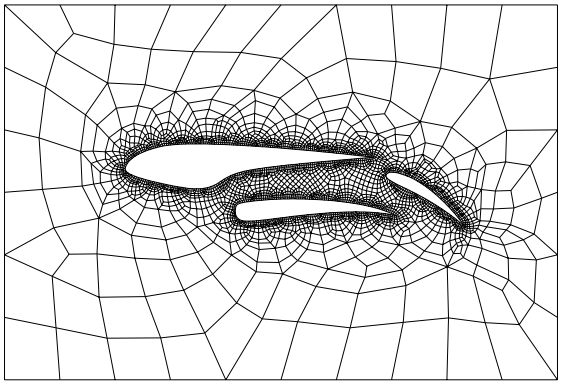
Figure 14. Example 3 (Continued)



(a) Initial mesh

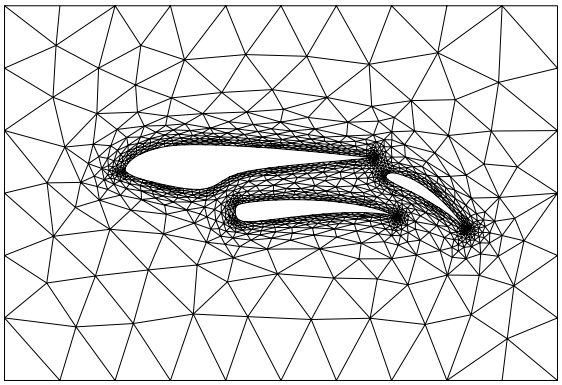


(b) Final triangular mesh, isotropic case

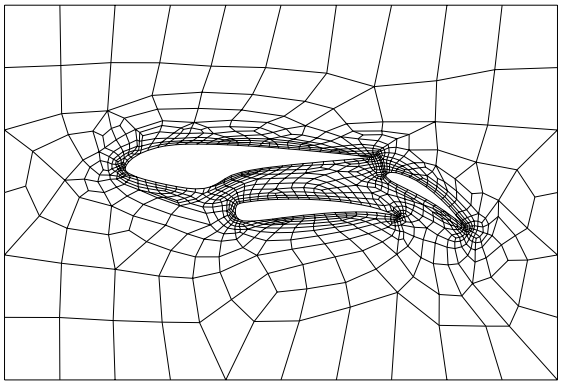


(c) Final quadrilateral mesh, isotropic case

Figure 15. Example 4



(d) Final triangular mesh, anisotropic case



(e) Final quadrilateral mesh, anisotropic case

Figure 15. Example 4 (Continued)

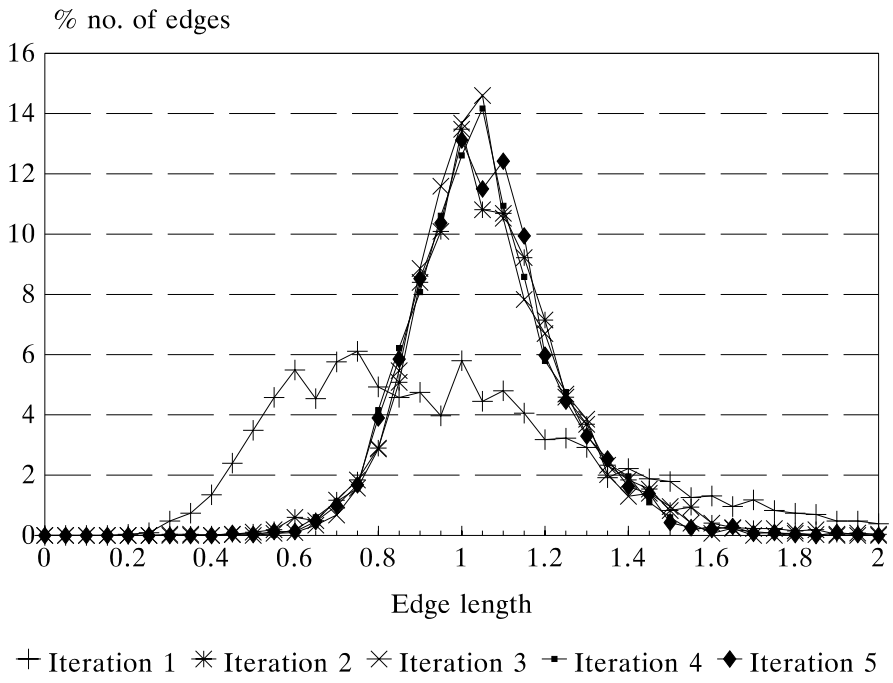


Figure 16. Convergence history for Example 4, triangular mesh, anisotropic case

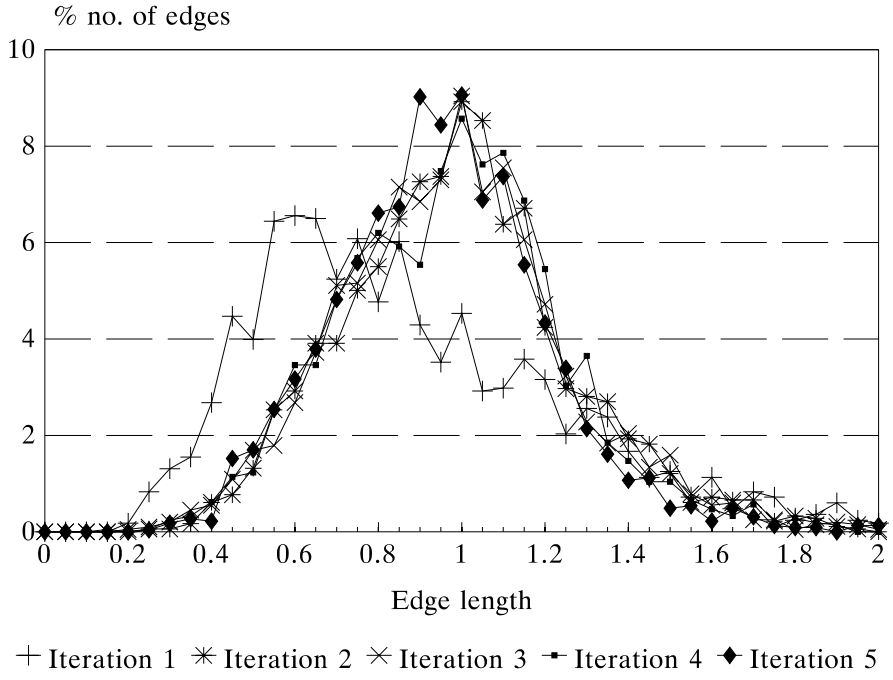


Figure 17. Convergence history for Example 4, quadrilateral mesh, anisotropic case