CHAPTER NINETEEN

Adaptive Methods

The ultimate goal in computational fluid dynamics is to obtain desired solutions as accurately as possible while minimizing the requirement for computational resources. Thus, we ask: how do we achieve both "accuracy" and "efficiency" at the same time? Often we exercise a compromise where we may choose to sacrifice some accuracy for the sake of expediting a solution, or vice versa. Does an acceptable compromise exist? These are the types of questions that typically enter the minds of the CFD practitioner before undertaking a major project.

Given a fixed computational method and limited computer resources, one is confronted with a decision as to which direction to follow. The most feasible approach under these restricted circumstances will be to seek the best computational grid arrangement which will lend itself to the best possible accuracy and maximum efficiency. Adaptive methods are designed to achieve both accuracy and efficiency, with mesh refinements provided selectively only where needed.

The basic concept for adaptive methods consists of providing mesh refinements for efficiency (cost reduction) as dictated by predetermined criteria. The criteria for mesh refinements and unrefinements (coarsening) are determined by error indicators. The error indicators are usually represented by gradients of a suitable variable – the larger the gradient, the finer the mesh required.

In line with the two different grid generation schemes, structured and unstructured, two different adaptive methods are available, one for structured grids and another for unstructured grids. The structured adaptive methods are presented in Section 19.1, with the unstructured adaptive methods in Section 19.2.

19.1 STRUCTURED ADAPTIVE METHODS

Structured adaptive meshes may be constructed either by a control function approach or by a variational function approach. We shall discuss both of these methods next.

19.1.1 CONTROL FUNCTION METHODS

19.1.1.1 Basic Theory

In this method, grid points are moved in accordance with weights or control functions reflecting the gradients of the variables, the process known as redistribution. Adaptive

redistribution of the points is based on the principle of equal distribution of error by which a point distribution is set so as to make the product of the spacing and a weight function W constant over the points. This idea is represented by

$$Wdx = \text{constant} \tag{19.1.1}$$

With the point distribution defined by a function $x(\xi)$, where ξ varies by a unit increment between points, the equal distribution principle can be expressed as

$$Wx_\xi d\xi = Wx_\xi = \text{constant}, \quad d\xi = 1 \tag{19.1.2}$$

This one-dimensional equation can be applied in each direction in an alternating fashion (in the spirit of ADI). However, a direct extension to multiple dimensions can be made in either of two ways: control function approach, or variational approach. In the control function methods, we combine the elliptic grid generation system with the equal distribution principle given by (19.1.2).

$$g^{ij}\mathbf{r}_{,ij} + P^i\mathbf{r}_{,i} = 0 \tag{19.1.3}$$

where g^{ij} are the elements of the contravariant metric tensor [Chung, 1996]:

$$g^{ij} = \frac{\partial \xi_i}{\partial x_m}\frac{\partial \xi_j}{\partial x_m} \tag{19.1.4}$$

These elements are more conveniently expressed computationally in terms of the elements of the covariant metric tensor g_{ij}:

$$g_{ij} = \frac{\partial x_m}{\partial \xi_i}\frac{\partial x_m}{\partial \xi_j} \tag{19.1.5}$$

which can be calculated directly. Thus

$$g^{ij} = \frac{1}{g}\frac{\partial g}{\partial g_{ij}} \tag{19.1.6}$$

where

$$g = |g_{ij}| \tag{19.1.7a}$$

or

$$g = \left|\frac{\partial x_m}{\partial \xi_i}\frac{\partial x_m}{\partial \xi_j}\right| \tag{19.1.7b}$$

with $\mathbf{r}$ being the cartesian coordinates and ξ_i the curvilinear coordinates. The $P^i = P_i$ denotes the control function which controls the spacing and orientation of the grid lines in the field.

The one-dimensional form of this system is

$$\frac{\partial^2 x}{\partial \xi^2} + P\frac{\partial x}{\partial \xi} = 0 \tag{19.1.8}$$

Differentiation of (19.1.2) yields

$$W\frac{\partial^2 x}{\partial \xi^2} + \frac{\partial W}{\partial \xi}\frac{\partial x}{\partial \xi} = 0 \tag{19.1.9}$$

It follows from (19.1.8) and (19.1.9) that

$$-P = \frac{\partial^2 x/\partial \xi^2}{\partial x/\partial \xi} = -\frac{\partial W/\partial \xi}{W} \tag{19.1.10}$$

from which the control function P can be taken as

$$P = \frac{1}{W}\frac{\partial W}{\partial \xi} \tag{19.1.11}$$

This may be extended to three-dimensional geometries as

$$P_i = \frac{1}{W}\frac{\partial W}{\partial \xi_i} \quad (i = 1, 2, 3) \tag{19.1.12a}$$

or

$$P_i = \frac{1}{W_{(i)}}\frac{g^{ij}}{g^{(ii)}}\frac{\partial W_{(i)}}{\partial \xi_j} \tag{19.1.12b}$$

where the latter version (19.1.12b) requires the weight functions to be specified in all three directions [Eiseman, 1987].

19.1.1.2 Weight Functions in One Dimension

As seen in (19.1.12), the effect of the weight function W is to reduce the point spacing function x_ξ if W is large. Therefore, the weight function should be set as some measure of the solution error or the solution variation. The simplest choice in one-dimensional problems is the solution gradient, that is,

$$W = u_x \tag{19.1.13}$$

Substituting (19.1.13) into (19.1.2) yields

$$u_x x_\xi = constant$$

or

$$u_\xi = constant$$

With the solution gradient used as a weight function, the point distribution can be adjusted in such a manner that the same change in the solution occurs over each grid, as illustrated in Figure 19.1.1a. This choice for the weight function has the disadvantage of making the spacing infinitely large when the solution is constant.

In contrast, consider the solution gradient in the form

$$W = \sqrt{1 + u_x^2} \tag{19.1.14}$$

An increment of arc length, ds, on the solution curve $u(x)$ is given by

$$ds^2 = dx^2 + du^2 = \left(1 + u_x^2\right) dx^2$$

so that this form of the weight function may be written as

$$W = s_x$$

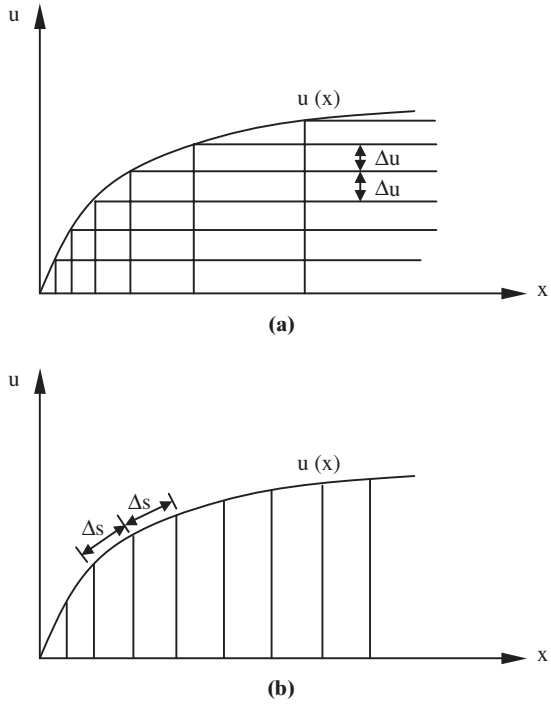


Figure 19.1.1 Relation between grid spacing and weight functions. **(a)** Constant solution gradient with $w = u_x$. **(b)** Constant solution gradient with $w = (1 + u_x^2)^{\frac{1}{2}}$.

which gives

$$s_x x_\xi = constant \tag{19.1.15a}$$

or

$$s_\xi = constant \tag{19.1.15b}$$

Thus, with the weight function defined by (19.1.14), the grid point distribution is such that the same increment in arc length on the solution curve occurs over each grid interval (Figure 19.1.1b).

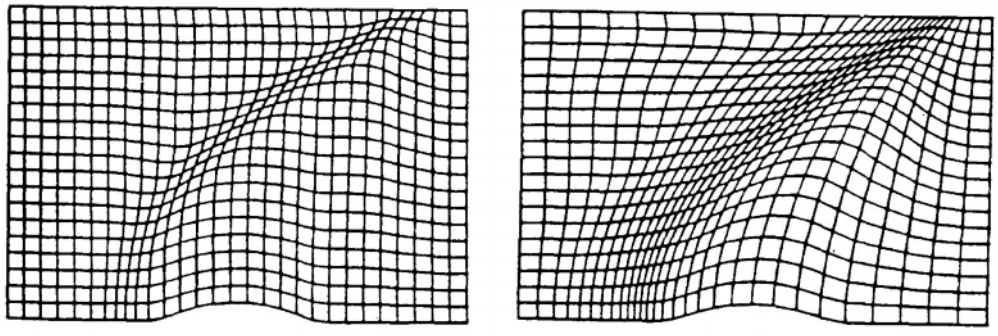
Unlike the previous choice, this weight function gives uniform spacing when the solution is constant. The concentration of points in the high-gradient region, however, is not as great.

In order to maintain desirable concentration of nodes at high gradient regions and peak solutions, the following weight function has been suggested [Eiseman, 1985]:

$$W = \left(1 + \alpha^2 u_x^2\right)^{\frac{1}{2}} (1 + \beta^2 |k|) \tag{19.1.16}$$

where α and β are user specified parameters and k is the curvature defined as

$$k = \frac{u_{xx}}{\left(1 + u_x^2\right)^{\frac{3}{2}}} \tag{19.1.17}$$



(a) Low α and β values (b) High α and β values

Figure 19.1.2 Effect of inclusion of curvature as well as gradient [Eiseman, 1985].

Here, large values of α and β contribute to closer nodal spacing, respectively, near the high solution gradient regions and solution extrema regions.

An alternative to (19.1.16) is to use

$$W = 1 + \alpha\,|u_x| + \beta\,|u_{xx}| \tag{19.1.18}$$

where α and β are non-negative parameters to be specified. An example based on (19.1.16) is shown in Figure 19.1.2a for low values of α and β and in Figure 19.1.2b for high values of α and β.

19.1.1.3 Weight Function in Multidimensions

The one-dimensional weight factors (19.1.13) based on the arc length on the solution curve can be generated to higher dimensions. Consider the position vector characterized by both the geometrical space **r** and solution space **u** such that

$$\mathbf{R} = \mathbf{r} + u\mathbf{e} = x_i\,\mathbf{i}_i + u\mathbf{e} \tag{19.1.19}$$

The covariant metric tensor is then given by

$$G_{ij} = \frac{\partial \mathbf{R}}{\partial \xi_i} \cdot \frac{\partial \mathbf{R}}{\partial \xi_j} = \left(\mathbf{r}_{\xi_i} + u_{\xi_i}\mathbf{e}\right) \cdot \left(\mathbf{r}_{\xi_j} + u_{\xi_j}\mathbf{e}\right)$$

or

$$G_{ij} = g_{ij} + u_{\xi_i} u_{\xi_j} \tag{19.1.20a}$$

where g_{ij} is the metric tensor in the physical space. Since

$$u_{\xi_i} = \nabla u \cdot \mathbf{r}_{\xi_i}$$

we obtain

$$G_{ij} = g_{ij} + \left(\nabla u \cdot \mathbf{r}_{\xi_i}\right)\left(\nabla u \cdot \mathbf{r}_{\xi_j}\right) \tag{19.1.20b}$$

and

$$|G_{ij}| = (1 + |\nabla u|^2)|g_{ij}| \tag{19.1.21}$$

In one dimension this reduces to the expression for arc length on the solution curve, that is,

$$\sqrt{G} = x_\xi \sqrt{1 + u_x^2} \tag{19.1.22}$$

In two dimensions, we have

$$\sqrt{|G_{ij}|} = \sqrt{g}(1 + |\nabla u|^2)^{\frac{1}{2}} \tag{19.1.23}$$

Thus, the extension of the one-dimensional weight function based on arc length on the solution curve to multidimensions is that based on area (2-D) or volume (3-D) on the solution curve,

$$W = (1 + |\nabla u|^2)^{\frac{1}{2}} \tag{19.1.24}$$

The weight functions as defined above can then be applied to the expressions for the control function given in (19.1.12a,b).

In multiple dimensions, adaptation should, in general, occur in all directions in a mutually dependent manner. If the solution varies only in one direction (say x) predominantly, then the adaptation may be carried out in that direction alone, using the one-dimensional weight function, with x replaced by the arc length along this line.

Examples of applications of the above schemes include Dwyer, Smook, and Kee [1982], Gnoffo [1980], and Nakamura [1982], among others.

19.1.2 VARIATIONAL METHODS

The classical theory of calculus of variations can be applied to problems requiring optimization or achieving the maximum degree of equal distribution of error [Brackbill and Saltzman, 1982]. With this in mind, we will examine the basic theory associated with equal distribution by means of variational methods.

19.1.2.1 Variational Formulation

The computational error can be reduced by distributing the grid points in such a way that the same positive weight function, $W(x)$, is equally distributed over the field as shown in Section 19.1.1. The nonuniform point distribution can be considered to be a transformation, $x(\xi)$, from a uniform grid in ξ-space, with the coordinate ξ serving to identify the grid points.

Let us now invoke a spring analogy so that, if the weight function $W(\xi)$ is a spring constant and x_ξ is the extension of the spring at ξ, then the energy stored in the spring is of the form

$$I = \frac{1}{2}\int_0^1 W(\xi)\, x_\xi^2 d\xi \tag{19.1.25}$$

It follows from the theory of calculus of variations that the integrand in (19.1.25) constitutes the "variational functional," F,

$$F(\xi,\ x,\ x_\xi) = \frac{1}{2} W(\xi)\, x_\xi^2 \tag{19.1.26}$$

and the energy stored in the spring, I, is known as the "variational principle."

There are two ways to obtain an optimum grid spacing:

(1) Minimization of the variational principle given by (19.1.25).
(2) Solving the differential equation(s) resulting from the so-called Euler-Lagrange equation,

$$\frac{\partial}{\partial \xi}\left[\frac{\partial F}{\partial(\partial x/\partial \xi)}\right] - \frac{\partial F}{\partial x} = 0 \tag{19.1.27}$$

Substituting (19.1.26) into (19.1.27) yields

$$\frac{\partial W}{\partial \xi}\frac{\partial x}{\partial \xi} + W\frac{\partial^2 x}{\partial \xi^2} = 0 \tag{19.1.28}$$

It is interesting to note that (19.1.28) is identical to (19.1.9), which originated from the general elliptic PDE representation (19.1.3).

The above process confirms the standard variational approach in which, given the differential equation [in this case the Poisson equation (19.1.28)], the corresponding variational functional (19.1.26) when substituted into the corresponding Euler-Lagrange equation (19.1.27) recovers the original differential equation (19.1.28).

This argument implies that, instead of using the PDE of the form (19.1.3), the variational approach suggests that, if there are means of obtaining many different forms of the variational functional, there will be a host of differential equations arising from this process, other than those of the standard form such as (19.1.3). The differential equations obtained in this manner are characterized by a variety of weight functions and subsequently the control functions leading to desired forms of adaptive procedures.

19.1.2.2 Smoothness Orthogonality and Concentration

To achieve adaptation with a maximum degree of smoothness, orthogonality, and desired concentration, our focus is to construct desirable forms of variational functionals [Brackbill and Saltzmann, 1982]. Toward this end, the following forms of variational principles are suggested:

(1) Smoothness

$$I_s = \int g^{ii}\, d\mathbf{x} \tag{19.1.29a}$$

$$I_s = \int \frac{1}{\sqrt{g}}(g_{ii}g_{jj} - g_{ij}g_{ij})d\xi \tag{19.1.29b}$$

(1) Orthogonality

$$I_o = \int g^{\frac{3}{2}} g^{ij} g^{ij}\, d\mathbf{x} \tag{19.1.30a}$$

$$I_o = \int (g_{ij}g_{ik} - g_{ii}g_{jk})(g_{mj}g_{mk} - g_{mn}g_{jk})d\xi \tag{19.1.30b}$$

(1) Concentration

$$I_W = \int W^2(\mathbf{x})\sqrt{g}d\mathbf{x} \tag{19.1.31a}$$

$$I_W = \int W^2(x)\, g\, d\xi \tag{19.1.31b}$$

The above formulation may be generalized using the directional control as follows [Brackbill, 1982]:

$$I(\xi) = \int_{\mathbf{x}} F(\xi)\, d\mathbf{x} \tag{19.1.32}$$

with

$$F(\xi) = \frac{1}{2} g^{ij}(\mathbf{x}) g_{\alpha\beta}(\xi) \frac{\partial \xi_\alpha}{\partial x_i} \frac{\partial \xi_\beta}{\partial x_j} \tag{19.1.33}$$

The Euler-Lagrange equation in general curvilinear coordinates takes the form

$$\left[g^{ij} \frac{\partial F}{\partial(\partial \xi_\alpha / \partial x_j)} \right]_{|i} - (g^{\alpha\lambda} F)_{|\alpha} = 0 \tag{19.1.34}$$

where the stroke " | " denotes the covariant derivative. Performing the covariant differentiation on (19.1.34) leads to

$$\frac{1}{\sqrt{g}} \frac{\partial}{\partial x_i} \left(g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} \sqrt{g} \right) + g^{ij} \Gamma^\lambda_{\alpha\beta} \frac{\partial \xi_\lambda}{\partial x_i} \frac{\partial \xi_\beta}{\partial x_j} = 0 \tag{19.1.35}$$

where

$$\Gamma^\lambda_{\alpha\beta} = \frac{1}{2} g^{\lambda\gamma} [g_{\gamma\alpha,\beta} + g_{\gamma\beta,\alpha} - g_{\alpha\beta,\gamma}]$$

It can be shown that the second term in (19.1.35) vanishes. Taking a variational derivative of (19.1.35) gives

$$\delta I = \int_{\mathbf{x}} \left[\frac{1}{\sqrt{g}} \frac{\partial}{\partial x_i} \left(g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} \sqrt{g} \right) \right] \delta\xi \, d\mathbf{x} = 0 \tag{19.1.36}$$

Integrating (19.1.36) by parts,

$$\delta I = \int_\Gamma g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} n_i \delta\xi_\lambda \, d\Gamma - \int_{\mathbf{x}} g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} \delta\xi_{\lambda,i} \, d\mathbf{x} = 0$$

or

$$\delta I = \delta \left(\int_{\mathbf{x}} \frac{1}{2} g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} \frac{\partial \xi_\lambda}{\partial x_i} \, d\mathbf{x} - \int_\Gamma g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} n_i \delta\xi_\lambda \, d\Gamma \right) = 0 \tag{19.1.37}$$

This provides the variational principle to be given in the form

$$I = \int_{\mathbf{x}} \frac{1}{2} g^{ij} \frac{\partial \xi_\lambda}{\partial x_j} \frac{\partial \xi_\lambda}{\partial x_i} \, d\mathbf{x} \tag{19.1.38}$$

with the Neumann boundary condition

$$S_\lambda = \frac{\partial \xi_\lambda}{\partial x_i} n_i \tag{19.1.39}$$

Thus, it follows that

$$I = \int_{\mathbf{x}} \frac{1}{W} \nabla \xi_i \cdot \nabla \xi_i d\mathbf{x}$$

$$\nabla \cdot \left(\frac{1}{W} \nabla \xi_i \right) = 0$$

$$\nabla^2 \xi_i = \frac{1}{W} \frac{\partial W}{\partial \xi_j} \frac{\partial \xi_j}{\partial x_k} \frac{\partial \xi_i}{\partial x_k} = \frac{1}{W} \frac{\partial W}{\partial \xi_j} g^{ij}$$

$$\nabla^2 \xi_k = -g^{ij} \mathbf{r}_{,ij} \cdot \nabla \xi_k = \frac{1}{W} g^{ik} \frac{\partial W}{\partial \xi_i}$$

$$\nabla \xi^i \cdot \mathbf{r}_{,j} = \delta_{ij}$$

and

$$g^{ij} \frac{\partial}{\partial \xi_i} \left(W \frac{\partial \mathbf{r}}{\partial \xi_j} \right) \cdot \nabla \xi_k = 0$$

Finally,

$$g^{ij} \frac{\partial}{\partial \xi_i} \left(W \frac{\partial \mathbf{r}}{\partial \xi_j} \right) = 0 \tag{19.1.40}$$

An adaptive grid with directional control can be constructed and the mesh alignment control variational principle takes the form

$$I_d = \int_{\mathbf{x}} \frac{1}{W} [(\mathbf{A} \times \nabla \xi_1)^2 + (\mathbf{B} \times \nabla \xi_2)^2 + (\mathbf{C} \times \nabla \xi_3)^2] \, d\mathbf{x} \tag{19.1.41}$$

where

$$(\mathbf{A} \times \nabla \xi_1)^2 = (\mathbf{A} \times \nabla \xi_1) \cdot (\mathbf{A} \times \nabla \xi_1), \quad \text{etc.}$$

Let

$$W = \frac{|\nabla U|}{|U|} \tag{19.1.42}$$

where U is the variable under consideration and choose $\mathbf{A}, \mathbf{B}, \mathbf{C}$ to be mutually orthogonal. This implies that

$$\mathbf{B} \times \mathbf{C} \parallel \frac{\partial \mathbf{r}}{\partial \xi_1}, \quad \mathbf{C} \times \mathbf{A} \parallel \frac{\partial \mathbf{r}}{\partial \xi_2}, \quad \mathbf{A} \times \mathbf{B} \parallel \frac{\partial \mathbf{r}}{\partial \xi_3}$$

Using the vector identity

$$\sqrt{g} g^{ij} = \frac{1}{W} (V_k V_k \delta^{ij} - V^i V^j)$$

we obtain the variational functional in the form

$$F(\xi) = \sqrt{g_{(A)}}\, g^{ij}_{(A)} \frac{\partial \xi_1}{\partial x_i}\frac{\partial \xi_1}{\partial x_j} + \sqrt{g_{(B)}}\, g^{ij}_{(B)} \frac{\partial \xi_2}{\partial x_i}\frac{\partial \xi_2}{\partial x_j} + \sqrt{g_{(C)}}\, g^{ij}_{(C)} \frac{\partial \xi_3}{\partial x_i}\frac{\partial \xi_3}{\partial x_j} \tag{19.1.43}$$

Substituting (19.1.43) into the Euler-Lagrange equation yields

$$\left\{ g^{ij}_{(A)} \left[\frac{\partial}{\partial \xi_i}\left(W \frac{\partial \mathbf{r}}{\partial \xi_j} \right) - W\left[\nabla(\mathbf{A}\cdot\mathbf{A}) - (\nabla\cdot\mathbf{A})\mathbf{A} \right] \right] \right\} \cdot \nabla\xi_1 = 0 \tag{19.1.44a}$$

$$\left\{ g^{ij}_{(B)} \left[\frac{\partial}{\partial \xi_i}\left(W \frac{\partial \mathbf{r}}{\partial \xi_j} \right) - W\left[\nabla(\mathbf{B}\cdot\mathbf{B}) - (\nabla\cdot\mathbf{B})\mathbf{B} \right] \right] \right\} \cdot \nabla\xi_2 = 0 \tag{19.1.44b}$$

$$\left\{ g^{ij}_{(C)} \left[\frac{\partial}{\partial \xi_i}\left(W \frac{\partial \mathbf{r}}{\partial \xi_j} \right) - W\left[\nabla(\mathbf{C}\cdot\mathbf{C}) - (\nabla\cdot\mathbf{C})\mathbf{C} \right] \right] \right\} \cdot \nabla\xi_3 = 0 \tag{19.1.44c}$$

with

$$g^{ij}_{(A)} = \mathbf{A}\cdot\mathbf{A}g^{ij} - (\mathbf{A}\cdot\nabla\xi_i)(\mathbf{A}\cdot\nabla\xi_j), \quad \text{etc.}$$

Here, if $\nabla\xi_1 \parallel \mathbf{A}$, the $\mathbf{g}(A) = 0$ and the variational functional $\mathbf{F}$ has no extremum with respect to ξ_1. Thus (19.1.44a) has no solution.

The above difficulty may be resolved by introducing "regularization." To this end, we divide each reference vector by its maximum magnitude and regulate the variational principle in the form

$$I = (1-\lambda)I_s + \lambda I_d \tag{19.1.45}$$

with $0 \le \lambda \le 1$, λ being the user specified constant. It is seen that an increase of λ leads to an increase of the alignment of the reference vectors.

The solution of (19.1.44) has been carried out for applications to the Kelvin-Helmholtz instability problem in a driven magnetic flowfield (Figure 19.1.3) [Brackbill, 1982]. The complexity of formulation of the governing equations is considered a major difficulty if one chooses to use the variational functional of the form given in (19.1.43).

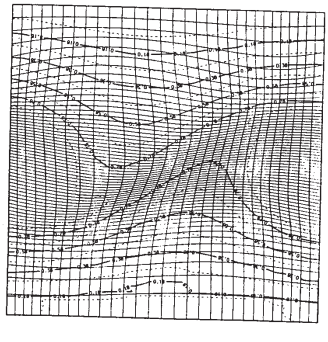


Figure 19.1.3 Magnetic flowfields based on direction control adaptive computational mesh [Brackbill, 1982].

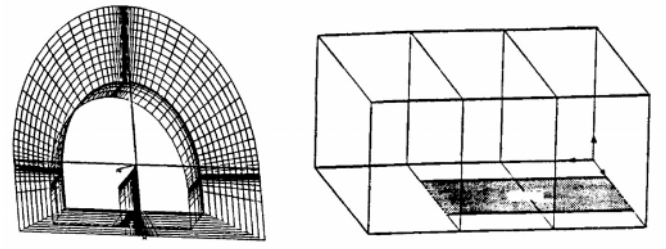


Figure 19.1.4 ONENA M6 wing multiblock [Kim and Thompson, 1990].

19.1.3 MULTIBLOCK ADAPTIVE STRUCTURED GRID GENERATION

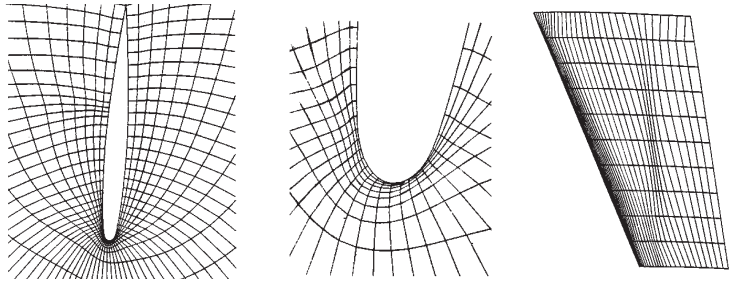
An adaptive structured grid mesh can be constructed in a multiblock environment, such as an aircraft wing shown in Figure 19.1.4 [Kim and Thompson, 1990]. Any block can be linked to any other block or to itself, with complete or lesser continuity across the block interfaces as specified by input. The adaptive control function with either an elliptic or a variational method is then formed at each point in the field by combining the interpolated components. For complex multiblock systems, the evaluation from the algebraic grid can be used.

In Figure 19.1.4, the three-initial grid is generated by the elliptic system with the $97 \times 17 \times 17$ grid divided into three blocks in the spanwise direction. The final grid adapted to the pressure gradients at the 50% span location and over the upper wing surface is shown in Figure 19.1.5a. The leading edge and upper surface details are shown in Figures 19.1.5b and 19.1.5c, respectively.

19.2 UNSTRUCTURED ADAPTIVE METHODS

For nearly two decades, unstructured adaptive methods have been extensively developed by Oden and his co-workers and Babuska and his co-workers. Some of their later works are summarized in Babuska et al. [1986] and Oden et al. [1986].

In the previous section, we presented structured adaptive methods in which an initially structured grid remains structured even after the adaptive process. In unstructured adaptive methods, however, we may begin with either a structured or unstructured grid system, but the final grid system, upon adaptation, becomes subsequently unstructured or remains unstructured, respectively.



(a) 50% span **(b)** Leading-edge detail **(c)** Upper surface

Figure 19.1.5 Final adaptive grid for Figure 23.3.1.

There are many different ways in which unstructured adaptive methods can be developed:

(1) mesh refinement methods (h-methods)
(2) mesh movement methods (r-methods)
(3) mesh enrichment methods (p-methods)
(4) combined mesh refinements and movements (hr-methods)
(5) combined mesh refinements and enrichments (hp-methods)

These methods will be elaborated in the next section.

19.2.1 MESH REFINEMENT METHODS (h-METHODS)

The basic concept of mesh refinement methods is to refine the element in which a posteriori error indicator (measure of error based on solution gradients) is larger than the preset criterion. This procedure is ideally applied to the finite element methods.

19.2.1.1 Error Indicators

In general, the solution error is not available a priori. Even after the solution has been completed "a posteriori error" must be evaluated from the so-called error indicator as the exact solution is not available. The a posteriori error indicator may be predicted from the solution gradients of variables: density, velocity, pressure, or temperature. For inviscid flows with shock waves, we may consider density gradients to be a best measure of error, whereas velocity gradients may play a key role in the case of compressible viscous flows. Pressure or temperature gradients can also be considered an important factor in determining the error indicator.

Let $\hat{u}$ and u be the exact solution and approximate solution, respectively. We then specify an error e in elliptic problems as

$$\|e\|_E = \|\hat{u} - u\|_E \leq \alpha \tag{19.2.1}$$

where $\|e\|_E$ represents the energy norm and α denotes a specific tolerance. We may rewrite (19.2.1) in the form

$$\|e\|_E \leq Csh \tag{19.2.2}$$

where C is a constant independent of mesh parameter h with $s = 1, 2, \ldots, n$ such that

$$(s-1)h \leq x \leq sh \tag{19.2.3}$$

The estimate given by (19.2.2) is known as a priori estimate based on some general information about the exact solution. A priori estimates indicate how fast the error changes as the h-refinements are changed. Estimates based on the approximate finite element solution are called a posteriori error estimates. To this end we write (19.2.1) in an alternate form.

$$\|e\|_E \leq \frac{k}{N^p} \tag{19.2.4}$$

where k and p are constants and N represents the number of degrees of freedom.

If p is the degree of polynomials for interpolations and (19.2.4) is extended to the h-adaptivity, then we have

$$\|e\|_E \leq \frac{k}{\exp(\beta N^\gamma)} \tag{19.2.5}$$

where k, β, and γ are positive constants. Taking a logarithm in (19.2.5) yields

$$\log \|e\|_E \leq \log\, k - \beta N^\gamma \log e \tag{19.2.6}$$

This represents the rate of convergence to be exponential [Babuska and Suri, 1990; Oden, Wu, and Legat, 1995]. If there are singular points in the domain, then the rate of convergence is algebraic.

To achieve the error estimate described above, the h-adaptivity proceeds with an error indicator, a dimensionless quantity, given in terms of ratios of gradients or rates of changes of gradients of appropriate variables. These variables may be density, velocity, pressure, or temperature. For example, density and velocity are usually chosen for the shock wave turbulent boundary layer flows.

The nondimensional error indicator is defined as

$$\theta = f(h,\ H^m), \quad m = 0,\ 1,\ 2 \tag{19.2.7}$$

where h is the mesh parameter and H^m is the Hilbert space (Section 8.3). For density and velocity as governing variables for determining the error indicator, we define, in terms of various semi-norms.

for density:

$$\theta = h\,|\rho|_{H^1}/|\rho|_{H^0}, \qquad \theta = h|\rho|_{H^2}/|\rho|_{H^1} \tag{19.2.8a}$$

for velocity:

$$\theta = h|\mathrm{v}_i|_{H^1}/|\mathrm{v}_i|_{H^0}, \qquad \theta = h|\mathrm{v}_i|_{H^2}/|\mathrm{v}_i|_{H^1} \tag{19.2.8b}$$

with

$$|\rho|_{H^0} = \left[\int_{\Omega_e} \rho^2\, d\Omega\right]^{\frac{1}{2}}, \qquad |\rho|_{H^1} = \left[\int_{\Omega_e} \frac{\partial\rho}{\partial x_i}\frac{\partial\rho}{\partial x_i}\, d\Omega\right]^{\frac{1}{2}},$$

$$|\rho|_{H^2} = \left[\int_{\Omega_e} \frac{\partial^2\rho}{\partial x_i \partial x_j}\frac{\partial^2\rho}{\partial x_i \partial x_j}\, d\Omega\right]^{\frac{1}{2}} \tag{19.2.9a}$$

$$|\mathrm{v}_i|_{H^0} = \left[\int_{\Omega_e} \mathrm{v}_i \mathrm{v}_i\, d\Omega\right]^{\frac{1}{2}}, \qquad |\mathrm{v}_i|_{H^1} = \left[\int_{\Omega_e} \frac{\partial \mathrm{v}_i}{\partial x_j}\frac{\partial \mathrm{v}_i}{\partial x_j}\, d\Omega\right]^{\frac{1}{2}},$$

$$|\mathrm{v}_i|_{H^2} = \left[\int_{\Omega_e} \frac{\partial^2 \mathrm{v}_i}{\partial x_j \partial x_j}\frac{\partial^2 \mathrm{v}_i}{\partial x_k \partial x_k}\, d\Omega\right]^{\frac{1}{2}} \tag{19.2.9b}$$

Here the mesh parameter h is determined for one-, two-, and three-dimensional elements as follows:

h = length of 1-D element

h = diameter of the circumcircle containing a 2-D element

h = diameter of the circumsphere containing a 3-D element

The order of the Hilbert space m may be increased to four if the fourth order biharmonic equations are to be solved.

It is often useful to rearrange the error indicator in terms of the energy norm E for any variable ϕ,

$$\theta = \left[\frac{h^2 |\nabla^2 \phi|}{h\,|\nabla \phi| + \varepsilon\,|\phi|} \right]_E \tag{19.2.10}$$

where ε is the computational noise control parameter [Löhner and Baum, 1990]. This criterion is particularly effective in shock wave discontinuities.

The FDV parameters discussed in Sections 6.5 and 13.6 can be used as the error indicator. Since the FDV parameters are calculated by changes in Mach number, Reynolds number, Peclet number, and Damköhler number, they represent more precise variations of the gradients of whichever variable(s) are dominant. Comparisons of the h refinements by various error indicators are shown in the following subsections.

19.2.1.2 Two-Dimensional Quadrilateral Element

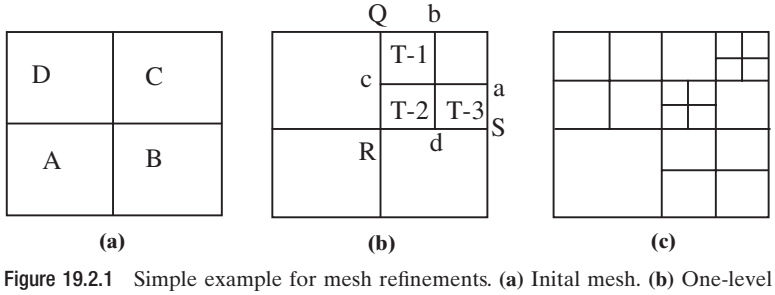
In Figure 19.2.1a, a simple example for two-level adaptation (refinements) is shown. Note that irregular nodes or hanging nodes arise in the process of refinements. For convenience, we shall permit only one irregular node along the side of unrefined element (nodes c and d, Figure 19.2.1b). For discretizations as shown in Figure 19.2.1c, however, the unrefined element D and B are subdivided even if not required by the error indicator.

In order that elements D and B remain linear in Figure 19.2.1b, we must eliminate nodes c and d as follows:

$$u_c = \frac{1}{2}(u_Q + u_R), \qquad u_d = \frac{1}{2}(u_R + u_S) \tag{19.2.11}$$

For the transition element, T1 we have

$$\begin{aligned} u^{(T1)} &= \Phi_1 u_1 + \Phi_2 u_2 + \Phi_3 u_3 + \Phi_4 u_4 \\ &= \Phi_1 \left(\frac{u_4 + u_R}{2} \right) + \Phi_2 u_2 + \Phi_3 u_3 + \Phi_4 u_4 \\ &= \begin{bmatrix} \Phi_1 & \Phi_2 & \Phi_3 & \Phi_4 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_R \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \end{aligned}$$



(a) (b) (c)

Figure 19.2.1 Simple example for mesh refinements. **(a)** Inital mesh. **(b)** One-level refinements. **(c)** Two-level refinements.

or

$$u^{(T1)} = \Phi_N H_{NM}^{(T1)} u_M^{(T1)} = \Phi_M^{(T1)} u_M^{(T1)} \tag{19.2.12}$$

with $\Phi_M^{(T1)} = \Phi_N H_{NM}^{(T1)}$ and $H_{NM}^{(T1)}$ being the interpolations and auxiliary matrix for the irregular element, respectively,

$$H_{NM}^{(T1)} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad \Phi_M^{(T1)} = [\Phi_1/2 \quad \Phi_2 \quad \Phi_3 \quad \Phi_1/2 + \Phi_4],$$

$$u_M^{(T1)} = [u_R \quad u_2 \quad u_3 \quad u_4]^T$$

Similarly,

$$u^{(T2)} = \Phi_N H_{NM}^{(T2)} u_M^{(T2)} = \Phi_M^{(T2)} u_M^{(T2)} \tag{19.2.13}$$

$$u^{(T3)} = \Phi_N H_{NM}^{(T3)} u_M^{(T3)} = \Phi_M^{(T3)} u_M^{(T3)} \tag{19.2.14}$$

with

$$H_{NM}^{(T2)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix}, \qquad u_M^{(T2)} = [u_1 \quad u_S \quad u_3 \quad u_Q]^T$$

$$H_{NM}^{(T3)} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad u_M^{(T3)} = [u_R \quad u_2 \quad u_3 \quad u_4]^T$$

In this manner the irregular nodes c and d in the first level refinements (Figure 19.2.1b) are eliminated and replaced by the global nodes Q, R, and S. Note that for two or higher level refinements, no new types of auxiliary matrix arise. This is because only one hanging node (irregular node) is to be allowed for any refinement process.

The procedure for h-refinement is as follows:

Step 1

A coarse finite element mesh is constructed, which contains only a small number of elements, sufficient to model basic geometrical features and flow characteristics. Obtain the preliminary flow solution on this initial mesh.

Step 2

Compute the error indicator for each element.

Step 3

If $\theta \geq \alpha E$, refine by subdividing the quadrilateral through midpoints. If $\theta \leq \beta E$, unrefine by reversing the refining process. Typically set $\alpha = 0.2$ and $\beta = 0.5$ with E being the user-specified tolerance.

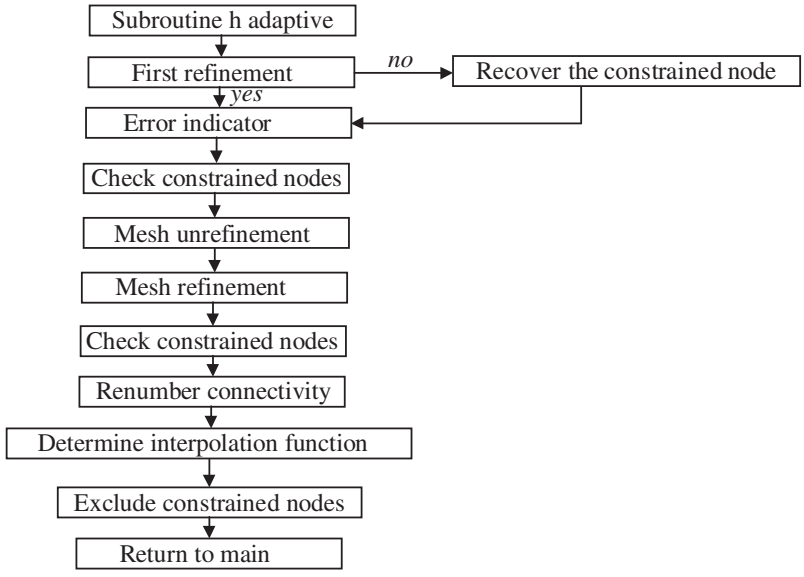


Figure 19.2.2 Flow diagram for mesh refinements.

The overall flow diagram is shown in Figure 19.2.2. Some applications to two-dimensional problems with triangular elements and quadrilateral elements using the error indicator (19.2.10) are shown in Figure 19.2.3 and Figure 19.2.4, respectively [Yoon and Chung, 1991].

Instead of using the primitive variable error indicators, we may take advantage of the FDV parameters as discussed in Section 13.6. To this end, we examine a compression

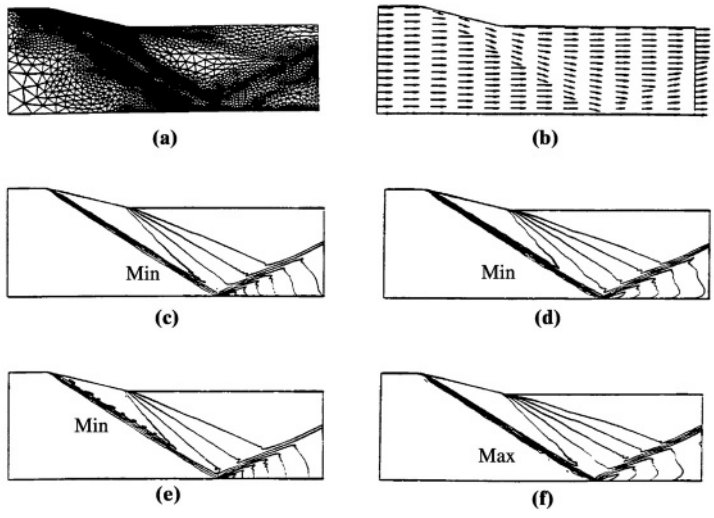


Figure 19.2.3 Mesh refinements (h-method), triangular elements. **(a)** Adapted mesh configuration (5004 elements, 2600 nodes, t = 0.059 sec). **(b)** Velocity field (Inlet vel. = 1506 m/sec). **(c)** Density contours (Max = 1.648, Min = 0.418, $\Delta = 0.123$ kg/m^3. **(d)** Temperature contours (Max = 1588, Min = 863, $\Delta = 73$K. **(e)** Pressure contours (Max = 0.725, Min = 0.115, $\Delta = 0.061$ MPa. **(f)** Mach number contours (Max = 2.603, Min = 1.152, $\Delta = 0.145$.

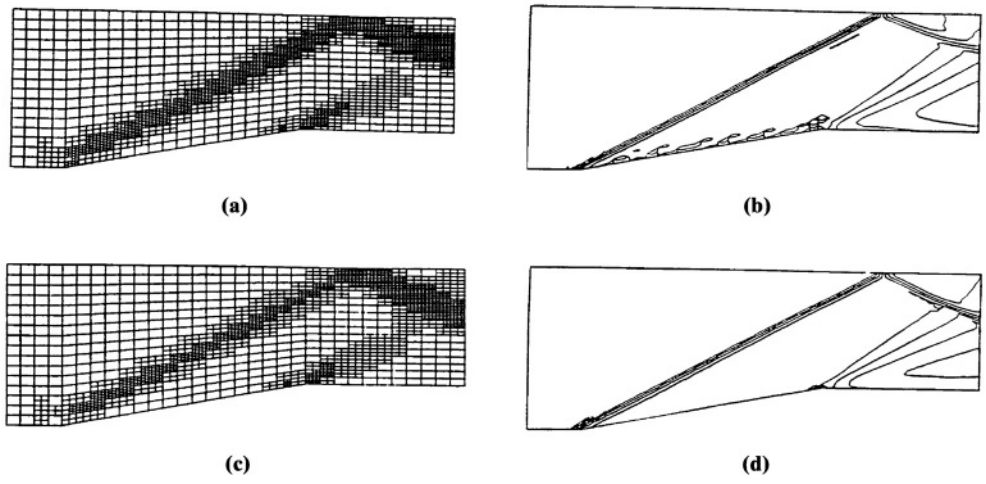


Figure 19.2.4 Mesh refinements (h-method), quadrilateral elements. **(a)** Adaptive mesh at t = 12 sec (2078 elements, 2317 nodes). **(b)** Mach number contours at t = 12 sec (max = 2.9, min = 1.4, $\Delta = 0.15$). **(c)** Adaptive mesh at t = 16 sec (1847 elements, 2078 nodes). **(d)** Mach number contours at t = 16 sec (max = 2.9, min = 1.3, $\Delta = 0.16$).

corner supersonic flow as shown in Figure 19.2.5a with initial grid of 4,600 elements (Figure 19.2.5b) [Heard and Chung, 2000]. Adaptative refinement is made for s_1 or s_3 greater than 0.45. Unrefinement is to be applied if the FDV parameters are less than 0.2. The contours of the first order FDV parameters (s_1, s_3) are shown in Figure 19.2.6. As demonstrated elsewhere (Section 6.6), these FDV parameters resemble closely the flowfield itself. Two level adaptation refinements have been carried out as shown in Figure 19.2.7. For the purpose of comparison, the results of computations based on the standard primitive variable error indicators are displayed. It is shown that the FDV results provide lesser number of adapted elements for both adaptative refinement levels. In addition, the refined regions are narrower for the FDV calculations. This is influenced by the FDV parameters being sensitive to the current flowfield physics dictating the decision for either refinement or unrefinement. These trends result in lesser computer time for the FDV-based error indicator. An additional advantage for the FDV approach is that the FDV parameters are already available in the formulation. The Mach number contours using the FDV error indicators and the primitive variable error indicators as shown in Figure 19.2.8 are practically identical.

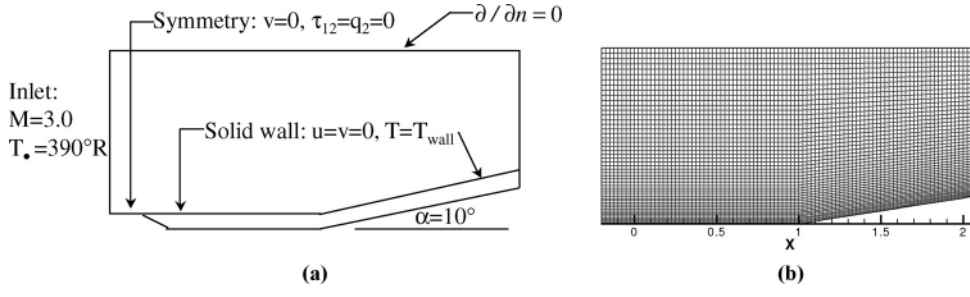


Figure 19.2.5 Geometries for adaptive mesh calculations for a compression corner supersonic flow using the flow field-dependent variation (FDV) parameters as error indicators. **(a)** Geometry and boundary conditions of compression corner. **(b)** Initial grid (4600 elements).

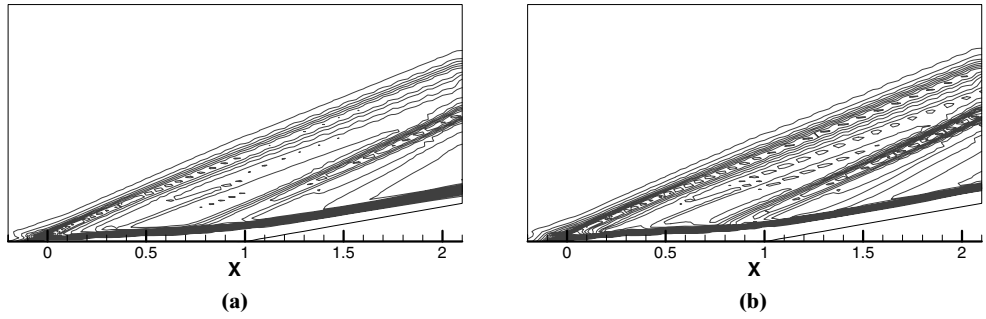


Figure 19.2.6 Contour plots of FDV parameters, resembling the flowfield for the compression corner flow of Figure 19.2.5. **(a)** First order convection FDV parameter (s_1). **(b)** First order diffusion FDV parameter (s_3).

19.2.1.3 Three-Dimensional Hexahedral Element

The refinement of three-dimensional hexahedral elements results in each hexahedral being divided into eight hexahedral elements, as shown in Figure 19.2.9a. During this refinement process, irregular, or hanging, nodes arise, similar to those produced during the refinement of two-dimensional quadrilateral elements. As demonstrated in the two-dimensional refinement process, only one irregular node is permitted along the side of an unrefined element. During this refinement process, elements may contain three, five, or six irregular nodes. The elimination process for these irregular nodes is demonstrated below.

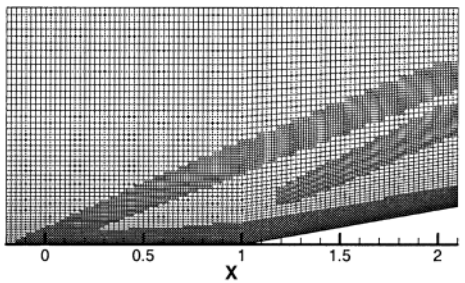
Three Irregular Nodes

Figure 19.2.9b shows the results of a simple one-level refinement of a hexahedral element. In the refinement process, three irregular or hanging nodes arise (nodes 1, 5, and 8). In order for the element I to remain linear, nodes 1, 5, and 8 must be eliminated as follows:

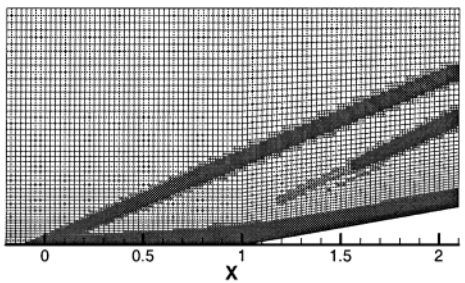
$$u_1 = \frac{1}{2}(u_4 + u_A), \qquad u_5 = \frac{1}{4}(u_A + u_4 + u_B + u_C), \qquad u_8 = \frac{1}{2}(u_4 + u_B)$$

Then, any flow property, u, is calculated from

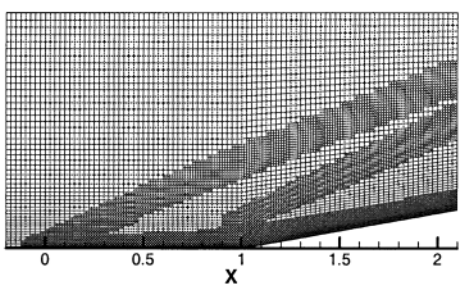
$$\begin{aligned} u &= \Phi_1 u_1 + \Phi_2 u_2 + \Phi_3 u_3 + \Phi_4 u_4 + \Phi_5 u_5 + \Phi_6 u_6 + \Phi_7 u_7 + \Phi_8 u_8 \\ &= \Phi_1\left(\frac{u_4 + u_A}{2}\right) + \Phi_2 u_2 + \Phi_3 u_3 + \Phi_4 u_4 + \Phi_5\left(\frac{u_A + u_4 + u_B + u_C}{4}\right) \\ &\quad + \Phi_6 u_6 + \Phi_7 u_7 + \Phi_8\left(\frac{u_4 + u_B}{2}\right) \\ &= \begin{bmatrix} \Phi_1 & \Phi_2 & \Phi_3 & \Phi_4 & \Phi_5 & \Phi_6 & \Phi_7 & \Phi_8 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} u_A \\ u_2 \\ u_3 \\ u_4 \\ u_C \\ u_6 \\ u_7 \\ u_B \end{bmatrix} \end{aligned}$$



(i) Adaptive mesh based on FDV parameters (9466 elements)

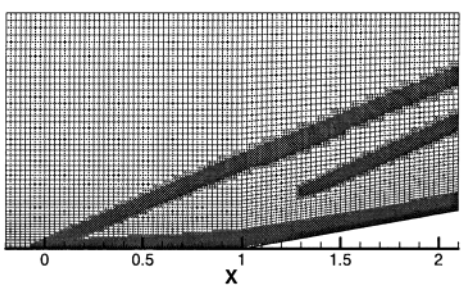


(i) Adaptive mesh based on FDV parameters (20836 elements)



(ii) Adaptive mesh based on primitive variable error indicators (9988 elements)

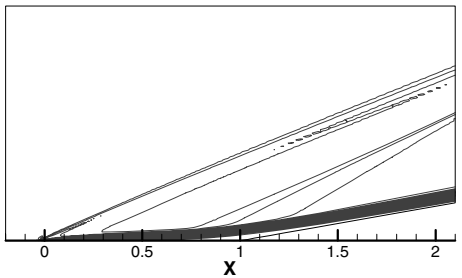
(a)



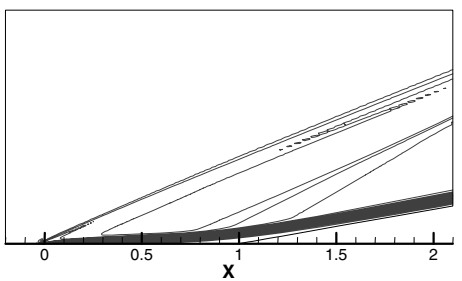
(ii) Adaptive mesh based on primitive variable error indicators (21346 elements)

(b)

Figure 19.2.7 Adaptive meshes for the first and second levels of refinement for the compression corner flow of Figure 19.2.5. **(a)** First grid refinement. **(b)** Second grid refinement.

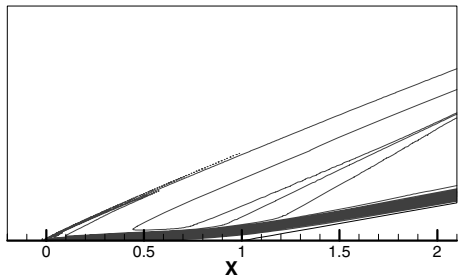


(i) Mach number contours calculated from FDV parameters

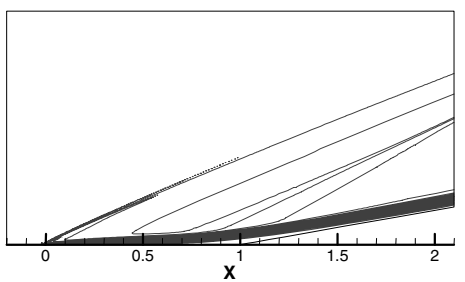


(ii) Mach number contours calculated from primitive variable error indicators

(a)



(i) Mach number contours calculated from FDV parameters



(ii) Mach number contours calculated from primitive variable error indicators

(b)

Figure 19.2.8 Mach number contours calculated from adaptive meshes, first and second levels of refinement using the error indicators based on the FDV parameters and primitive variable error indicators. **(a)** Mach number contours (first grid refinement). **(b)** Mach number contours (second refinement).

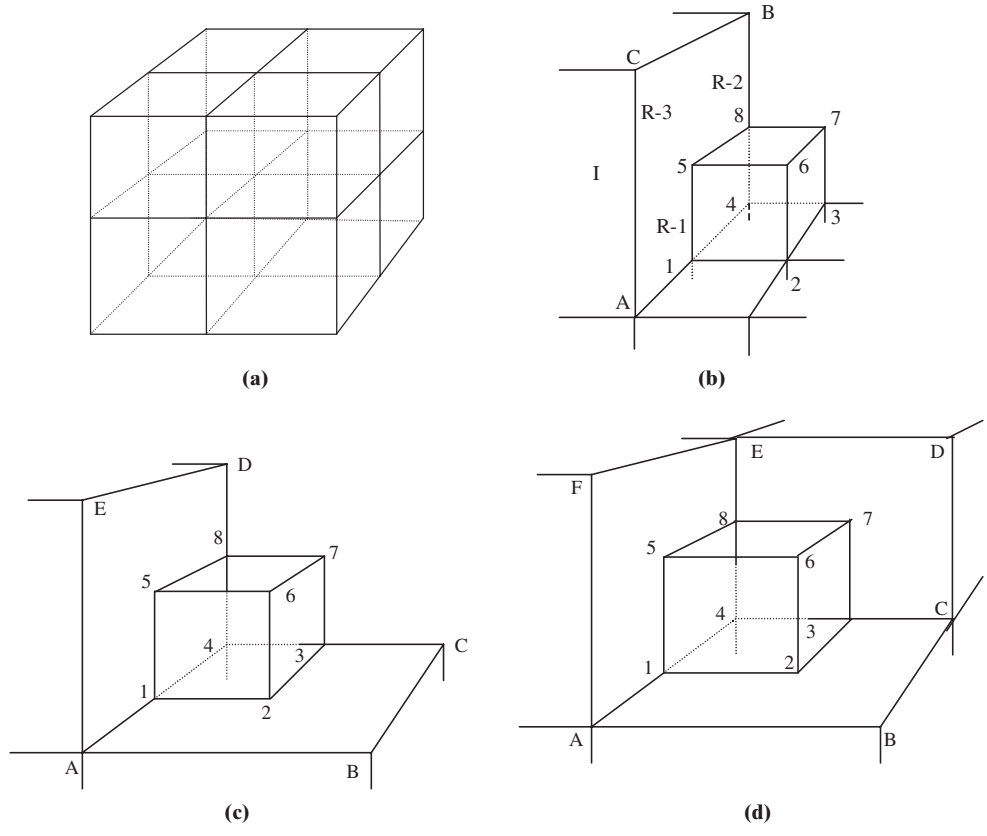


Figure 19.2.9 Mesh refinement of hexahedral element with hanging nodes. **(a)** Hexahedral elements. **(b)** Three hanging nodes. **(c)** Five hanging nodes. **(d)** Six hanging nodes.

or

$$u^{(T)} = \Phi_N H_{NM}^{(T)} u_M^{(T)}$$

with $\Phi_M^{(T)} = \Phi_N H_{NM}^{(T)}$ and $H_{NM}^{(T)}$ being the interpolation and auxiliarymatrix for the irregular node, respectively.

$$H_{NM}^{(T)} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \qquad u_M^{(T)} = \begin{bmatrix} u_A \\ u_2 \\ u_3 \\ u_4 \\ u_C \\ u_6 \\ u_7 \\ u_B \end{bmatrix}$$

$$\Phi_N^{(T)} = \left[\frac{\Phi_1}{2} + \frac{\Phi_5}{4} \quad \Phi_2 \quad \Phi_3 \quad \frac{\Phi_1}{2} + \Phi_4 + \frac{\Phi_5}{4} + \frac{\Phi_8}{2} \frac{\Phi_5}{4} \quad \Phi_6 \quad \Phi_7 \quad \frac{\Phi_5}{4} + \frac{\Phi_8}{2} \right]$$

It can be shown that similar auxiliary matrices are derived for the refined elements in locations R-1 through R-3 on element I, as well as for refined elements on the six other faces of element I, for a total of twenty-four auxiliary matrices.

Five Hanging Nodes

The refined element shown in Figure 19.2.9c contains five irregular nodes (nodes 1, 2, 3, 5, and 8). The auxiliary matrix for the irregular element can be derived similar to the procedure demonstrated above for three irregular nodes as shown below.

$$H_{NM}^{(T)} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \qquad u_M^{(T)} = \begin{bmatrix} u_A \\ u_B \\ u_C \\ u_4 \\ u_E \\ u_6 \\ u_7 \\ u_D \end{bmatrix}$$

Additional auxiliary matrices are derived, in accordance with the refined elements' location on the faces of the unrefined element. As in the case of three irregular nodes, the irregular elements can occupy four locations on each of six faces, for a total of twenty-four auxiliary matrices.

Six Hanging Nodes

Similarly, the refined element in Figure 19.2.9d contains six irregular, or hanging, nodes (nodes 1, 2, 3, 5, 7, and 8). The auxiliary transition matrix for the irregular element shown in Figure 19.2.9 is

$$H_{NM}^{(T)} = \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{4} & \frac{1}{4} & 0 & 0 & \frac{1}{4} & \frac{1}{4} \\ 0 & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix} \qquad u_M^{(T)} = \begin{bmatrix} u_A \\ u_B \\ u_C \\ u_4 \\ u_F \\ u_6 \\ u_D \\ u_E \end{bmatrix}$$

There are also twenty-four possible auxiliary matrices for elements with six irregular nodes. However, in practice, it is not necessary to store all possible auxiliary matrices in

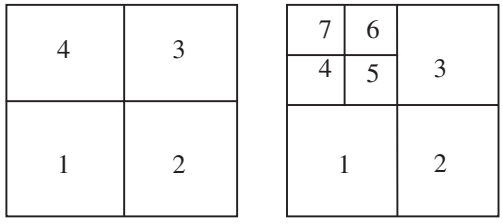


Figure 19.2.10 Illustration of refinement/unrefinement process. **(a)** Initial mesh. **(b)** Adapted refined mesh.

computer memory. Note that the rows of the auxiliary matrix for an irregular element can be determined simply by noting the number of nodes required for interpolation at the irregular node, and the node numbers used for interpolation. The nonzero entries are placed in the columns corresponding to the node numbers used for interpolation. Those nonzero entries are either $1/2$ or $1/4$, depending on whether the value is interpolated between two or four nodes.

Mesh Unrefinement

For time dependent problems, in which a discontinuity may be migrating over the grid during a given time interval, an unrefinement procedure is needed, as well as a refinement procedure. In a similar way as elements are refined if the error is greater than a specified number, elements are unrefined if the error is less than a specified number. One method of unrefinement is to require that only groups of elements that were refined before can be unrefined.

To ensure that the unrefinement occurs in the same way as the refinement, an array is added to the data structure to reflect the refinement history of the mesh [Devloo, Oden, and Pattani, 1988]. Toward this end, we introduce an array, NELGRP, in which NELGRP(I,IGR) is the Ith element of group IGR. If NELGRP(I,IGR) > 0, it refers to an element; if NELGRP(I,IGR) < 0, it refers to a group.

Using this array, we build a data structure of groups that contain elements or groups. When an element is refined, it is transformed into a group which contains four (or eight, for 3-D hexahedral elements) new subelements. All references to the original element are changed to the new group. For example, consider the simple grid of four elements in Figure 19.2.10a. If element 4 is refined as shown in Figure 19.2.10b, the NELGRP array for group 4 becomes:

$$\text{NELGRP}(1,4) = 4, \quad \text{NELGRP}(2,4) = 5, \quad \text{NELGRP}(3,4) = 6, \text{ and}$$
$$\text{NELGRP}(4,4) = 7$$

If element 5 were further refined, then NELGRP(2,4) would be changed to -5, referring to group 5, and the entries for NELGRP(,5) would be filled in with the new element numbers. Using this data structure, it becomes a simple matter of unrefining a group to recover the previous elements, when the error indicates unrefinement is needed.

19.2.2 MESH MOVEMENT METHODS (*r*-METHODS)

Instead of refining elements, grid points can be moved around (mesh redistribution) to provide clustering in certain regions by means of error indicators, known as the position

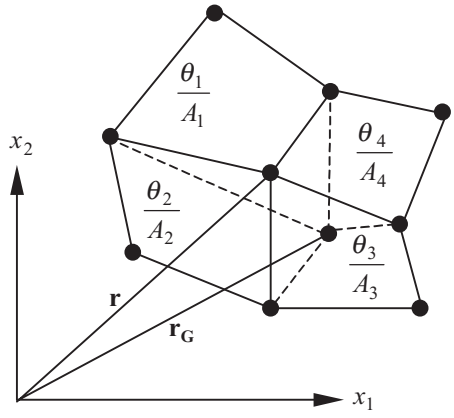


Figure 19.2.11 Equidistribution of element error indicators in a cluster of four elements.

vector approach [Oden, 1988]. This is in contrast to the clustering obtained by means of control functions in structured grids. Mesh movements may be combined with mesh refinements, known as the combined mesh refinement and mesh movement methods (*hr*-methods).

If the number of grid points is fixed, then it is desirable to relocate the nodes so that errors are equi-distributed over the entire mesh, in other words, the error over each element is the same. This method is often called the "moving mesh method." The procedure is as follows:

(1) Generate an initial mesh and obtain a trial solution.
(2) Compute the error indicator, θ.
(3) Determine the area weighted quantity θ_i / A_i for the element i (Figure 19.2.11).
(4) Let $\mathbf{r}_o$ be a position vector from the origin of a global coordinate system to the controlled element e_i of group G. The controlled element of the error of group G is

$$\mathbf{r}_G = \frac{\sum_{i=1}^{n} \mathbf{r}_i \theta_i / A_i}{\sum_{i=1}^{n} \theta_i / A_i} \tag{19.2.15}$$

where n denotes the number of elements surrounding a global node.
(5) Relocate the node at the centroid of group G to lie at the vertex r_G.
(6) Continue this process over each group G of n elements until the new location of each node remains within a prescribed tolerance.

An example of the r-method is shown in Figures 19.2.12, with applications demonstrated in Figure 19.2.13. The same approach can be used for triangular elements, and any number of surrounding elements can be accommodated at a global node under consideration.

19.2.3 COMBINED MESH REFINEMENT AND MESH MOVEMENT METHODS (*hr*-METHODS)

In this approach, the mesh is simultaneously refined and moved around as dictated by the error indicator. This adaptive process is implemented most conveniently in conjunction

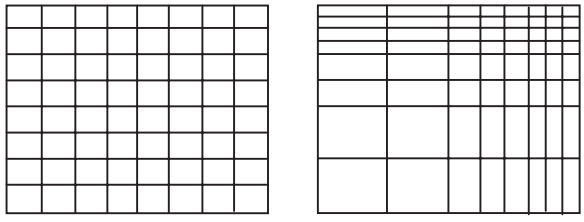


Figure 19.2.12 Example of *r*-methods. **(a)** Initial mesh. **(b)** Mesh after *r*-adaptivity.

with AFM or DVM mesh generators. There are two types of approaches in *hr*-methods: (1) mesh stretching and (2) local remeshing, described below.

Mesh Stretching

In this method, the measure of the error of each element is expressed in one dimension as

$$h_e^2 \left| \frac{d^2\rho}{dx^2} \right|_e = const \tag{19.2.16a}$$

With the second derivative of density evaluated at each node P or the current mesh, the new mesh may be generated with local spacing δ_P such that (Figure 19.2.14):

$$\delta_P^2 \left| \frac{d^2\rho}{dx^2} \right|_P = const \tag{19.2.16b}$$

For two dimensions with the local principal direction x_1 (major) and x_2 (minor),

$$\delta_{(1)P}^2 \left| \frac{d^2\rho}{dx_1^2} \right|_P = \delta_{(2)P}^2 \left| \frac{d^2\rho}{dx_2^2} \right|_P = \delta_{\min}^2 \left| \frac{d^2\rho}{dx_1^2} \right| = const \tag{19.2.17}$$

with $\partial^2\rho/\partial x_1^2 > \partial^2\rho/\partial x_2^2$ and $\delta_{(1)P}$ and $\delta_{(2)P}$ denoting node spacings in the x_1 and x_2 directions, respectively. Here $|\partial^2\rho/\partial x_1^2|_{\max}$ is the maximum value of $|\partial^2\rho/\partial x_1^2|_P$ over each node in the current mesh and $\delta_{\min}$ is a user-specified minimum value for δ in the new mesh. Thus, the local stretching parameter S_P is defined as

$$S_P = \sqrt{\left| \frac{d^2\rho}{dx_1^2} \right|_P \Big/ \left| \frac{d^2\rho}{dx_2^2} \right|_P} \tag{19.2.18}$$

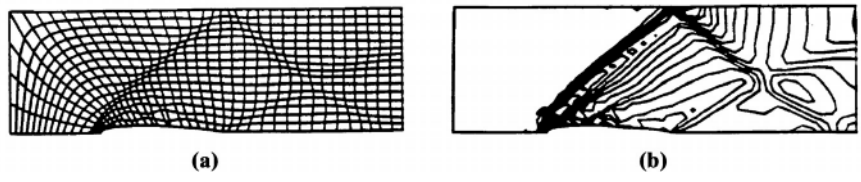


Figure 19.2.13 Example of an *r*-method for NACA 0012 airfoil in supersonic wind tunnel. **(a)** Mesh redistributions (10 applications). **(b)** Density contours.

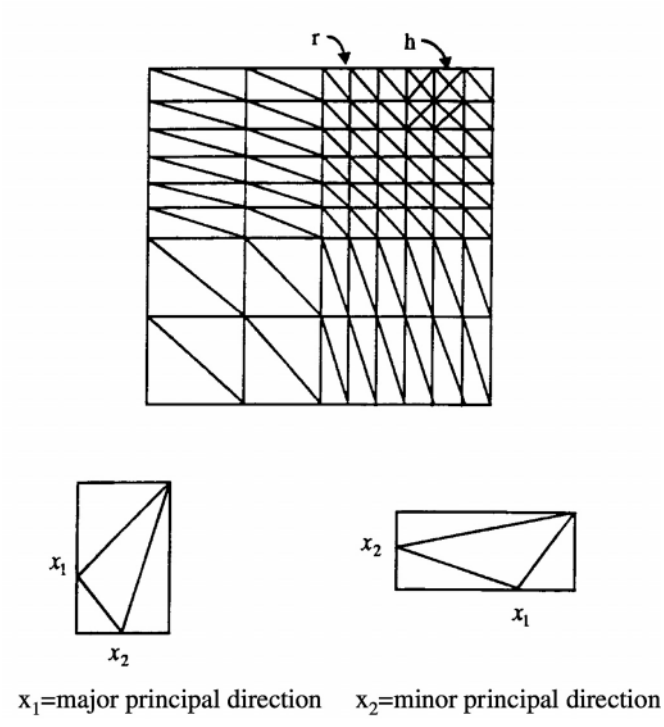


Figure 19.2.14 Example of mesh stretching scheme of *h*-method.

If δ_P computed from (19.2.17) is larger than the user-specified value δ_{max}, then we set $\delta_P = \delta_{max}$. Similarly, the node spacing will be controlled such that $\delta_P = \delta_{max}$ (user-specified maximum allowable spacing). It is thus expected from (19.2.18) that high stretching occurs only in the vicinity of one-dimensional flow features with low curvature.

In this manner, the mesh is regenerated in accordance with computed distribution of the mesh parameters and the solution of the problem recomputed on the new mesh. Obviously, the δ_{min} chosen governs the number of elements in the new mesh. This process continues until an acceptable quality of solution is achieved.

An example of a regular shock reflection at a wall with the sequence of remeshing is shown in Figure 19.2.15 [Peraire et al., 1987]. This method is prone to an excessive stretching, which is often an undesirable consequence.

Local Remeshing

To circumvent the excessive stretching, local remeshing may be employed. In this approach [Probert et al., 1991], a block element having large errors is removed and remeshed with fine mesh. Here the initial mesh is marked for deletion, new boundary points are generated, and triangulation is processed with the current front in conjunction with AFM.

Some applications for a shock tube and indentation flowfields are shown in Figure 19.2.16a and Figure 19.2.16b, respectively [Probert et al., 1991].

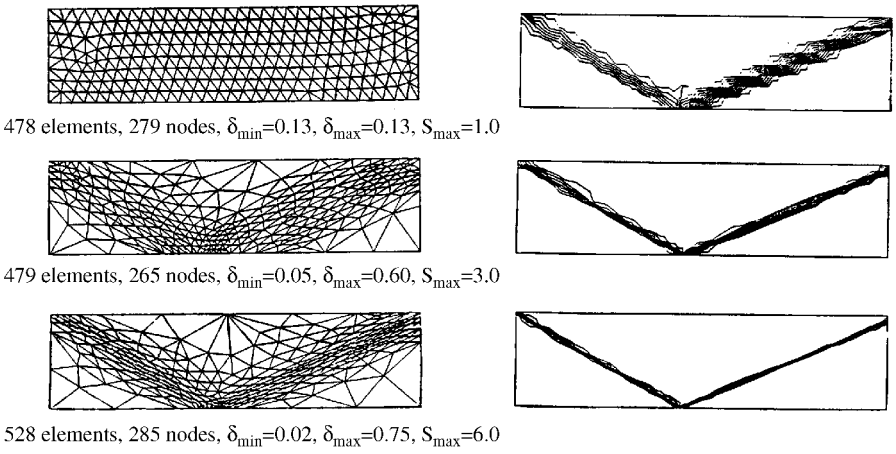


Figure 19.2.15 Local remeshing process for regular shock reflection at a wall and corresponding flowfields [Peraire et al., 1987].

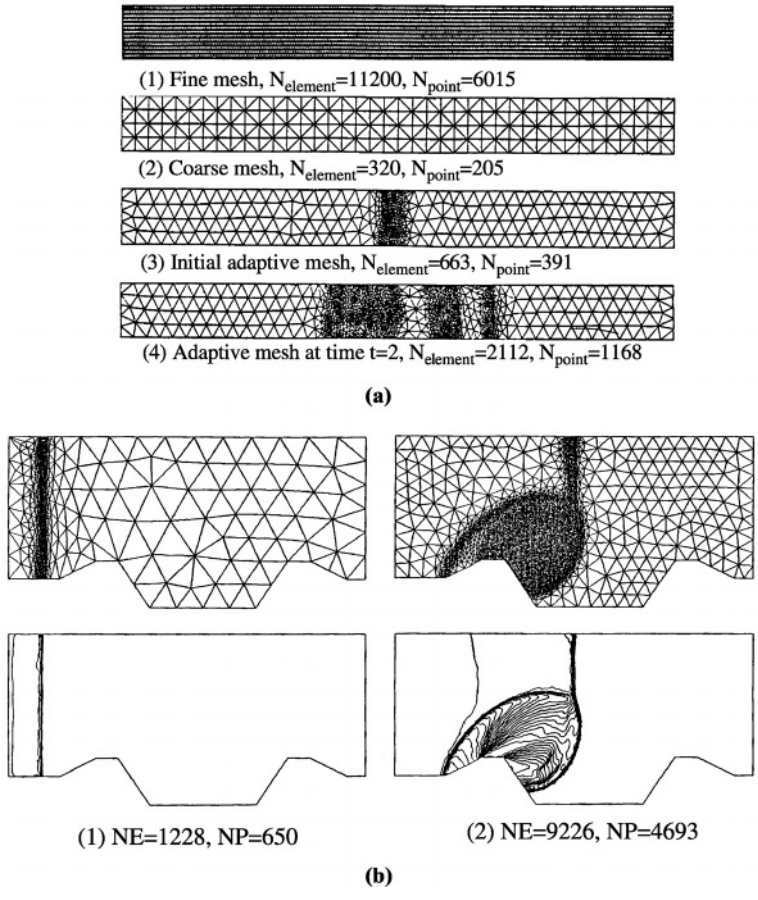


Figure 19.2.16 Local remeshing with AFM [Probert et al., 1991]. **(a)** Propagation of a planar shock. **(b)** Computation of the flow field produced by a strong shock passing over an indentation showing the mesh and corresponding density contours at four different times.

19.2.4 MESH ENRICHMENT METHODS (*p*-METHODS)

This is the fundamental concept employed in finite element methods. Given a fixed mesh, improved solutions are expected to be achieved with an increase in the degree of the polynomials, or higher order approximations.

In this section, we are concerned with hierarchical interpolation function or the so-called *p*-version finite element approximation functions. The use of hierarchical interpolations was the focus of discussion in the spectral element methods in Section 14.1. Our attention here, however, is to seek adaptivity as required by the error indicator, resulting in various degrees of polynomials for different elements.

A need for increasing the degree of an approximation while keeping mesh sizes fixed is particularly important when boundary layers or singularities are encountered. One approach is to construct a hierarchical interpolation system in the form

$$\mathbf{U} = \Phi_\alpha \mathbf{U}_\alpha + \Phi_r^{(E)} \hat{\mathbf{U}}_r + \Phi_{rs}^{(F)} \hat{\mathbf{U}}_{rs} + \Phi_{rst}^{(I)} \hat{\mathbf{U}}_{rst} \tag{19.2.19}$$

for 3-D domain, similarly as in (14.1.16) with each function representing the tensor products of chosen polynomials (Chebyshev, Legendre, Lagrange, etc.). The degree p will be raised as required when the user-specified error indicator tolerance is exceeded. The hierarchical interpolation system (19.2.19) was detailed in Section 14.1.2 for the spectral element methods.

Recall that no side or interior nodes are installed physically (Figure 14.1.1), but higher order modes corresponding to the sides and interior are combined with the corner nodes. By means of static condensation, all side and interior mode variables are squeezed out of the final algebraic equations. This process allows the side and interior mode variables acting as the source terms, which are explicitly calculated.

In order to treat adjacent elements in which degrees of approximations are different as a result of adaptivity, special procedures are developed between the constrained and unconstrained nodes in the approach of Oden and co-workers [1989]. In such a procedure, the so-called constrained matrices are derived so that compatibility between two elements with differing degrees of approximations can be ensured. It is obvious that this is not necessary in the method of spectral elements as shown in Section 14.1. This is because whatever the Legendre polynomial orders of approximations, the final form of the element matrix is transformed into a linear isoparametric interpolation in terms of only the corner nodes. In this process, no side, edge, surface, of interior nodes are required. The higher order spectral approximations are represented only through summation of nodes, not associated with any physically assigned non-corner nodes.

Implementation of the *p*-method is seen to be identical to that of the spectral element methods, except that varying degrees of spectral orders can be employed for each element as dictated by error indicators. If any element fails to pass the predetermined (user-specified) tolerance requirement as judged from the calculated error indicator, the spectral order for this element must be raised. Then, along the boundaries (sides, edges, faces) of adjacent elements, there exist differences in degrees of freedom. In this case, we set the higher order element to dictate the degrees of freedom along the adjoining boundary. Other than the adaptive procedure, details of formulations for *p*-methods are identical to the SEM of Section 14.1.

19.2.5 COMBINED MESH REFINEMENT AND MESH ENRICHMENT METHODS (*hp*-METHODS)

If shock waves are interacting with (turbulent) boundary layers, the *p*-method alone is not adequate. Shock wave discontinuities can best be resolved through mesh refinements, and it is thus necessary that mesh enrichments which are efficient for boundary layers be combined with mesh refinements.

The simplest approach in this case is that the *h*-method is applied with only corner nodes of isoparametric elements until the shock waves are captured. Then we employ the *p*-version process with Legendre polynomials for boundary layer resolutions. This combined operation is to continue until all error indicator criteria are satisfied, with density and velocity gradients, respectively, being used for the *h*-version (shock waves) and *p*-version (boundary layers).The *hp* methods have been studied extensively by Babuska and his co-workers [1986–1998] and Oden and his co-workers [1986–1998].

In the process of adaptation, as dictated by the error indicator, a decision has to be made at any stage, whether *h*-refinements or *p*-enrichments are to be performed. One approach is to begin with low order polynomials and continue until *h*-refinements reach a certain level (for example, shock discontinuities have been resolved), followed by *p*-enrichments which are designed for resolving turbulence microscales such as in wall boundary layers or free shear layers. Another option is to rely on an optimization process in which an automatic decision is made as to whether *h*-refinements or *p*-enrichments are more desirable at any given stage of adaptation.

In the *hp* adaptivity, the error estimates and error indicators discussed in the *h*-version and *p*-version are combined. For a particular mesh and *p*-distribution, however, it is not possible to predict the accuracy a priori. Thus, we must rely on a posteriori error estimates using the finite element solutions.

To this end, we consider any function $u \in H^r(k)$ and a sequence of interpolations w^{hp} such that for any $0 \leq s \leq r$, and polynomial of degree $\leq P_k$

$$\|u - w^{hp}\|_{s,k} \leq \frac{c\, h_k^{\mu - s}}{P_k^{r-s}} \|u\|_{r,k}, \quad P_k = 1,\ 2,\ \ldots \tag{19.2.20}$$

with

$$\mu = \min(P_k + 1\,,\ r) \tag{19.2.21}$$

This is the error estimate applicable for the *hp* process [Babuska and Suri, 1990; Oden et al., 1995], with the error indicator given by

$$\theta = \frac{h_k}{P_k} |u|_k, \quad r = 2 < P + 1 \tag{19.2.22}$$

In practice the error indicator can be determined using the element residual technique. The fine mesh is obtained by raising the order of approximation by one for each node uniformly throughout the mesh. Then for each element *k*, the added shape function is interpolated in the sense of *hp* interpolation using the old shape functions. By subtracting the interpolates from each of the added shape functions, we effectively construct a basis for the element space of bubble function (Legendre polynomials, Chebyshev polynomials, Lagrange polynomials, etc.). The constrained approximation is fully taken

into account. Next, the local problems are formulated and solved and the element error indicators are calculated using the gradients of variables as shown in (19.2.1) through (19.2.9).

A typical adaptive *hp*-method based on the error estimate proceeds as follows:

(1) Input initial data, global tolerance E_G, and local tolerance $E_L < E_G$.
(2) Solve the problem on the current finite element mesh.
(3) For each element k in the mesh, calculate the error indicator θ_k, if $\theta_k > E_L$, then refine the element.
(4) Calculate the global estimate

$$\theta_G = \sqrt{\sum_k \theta_k^2} \tag{19.2.23}$$

If $\theta_G > E_G$ then decrease the local tolerance $E_L = 90\%\ E_G$, go to (2).

In order to estimate the local quality of an error estimate, we introduce the local effectivity index γ_k:

$$\gamma_k = \frac{\theta_k}{\|e\|_k} \tag{19.2.24}$$

Introducing a discrete measure (weight) w_k

$$w_k = \frac{\|e\|_k^2}{\|e\|^2} \tag{19.2.25}$$

we obtain

$$\gamma^2 = \sum_k \gamma_k^2 w_k \tag{19.2.26}$$

Thus, the global effectivity index (squared) can be interpreted as the average of the local indices (square) weighted with respect to the discrete measure; more emphasis is placed upon elements with large errors and less on elements for which the error is small.

We may utilize the notion of standard deviation σ as a quantity estimating the discrepancy of the local effectivity indices.

$$\sigma^2 = \sum_k \left(\gamma_k^2 - \gamma^2\right)^2 w_k \tag{19.2.27}$$

This can be normalized to

$$\overline{\sigma}^2 = \sum_k \left(\overline{\gamma}_k^2 - 1\right)^2 w_k \tag{19.2.28}$$

with

$$\overline{\gamma}_k = \frac{\theta_k}{\|e\|\gamma - 1} \tag{19.2.29}$$

Equation (19.2.28) may be used as a criterion to compare the quality of various error estimates.

Our objective in the *hp*-method is to optimize the distribution of mesh size h and polynomial degree p over a finite element. For given h-refinements, the p-distributions may vary from element to element, as shown in Figure 14.1.2. Notice that boundaries between the higher and lower p's are dictated by the higher degrees polynomial with irregular nodes and elements treated as discussed in Section 19.2.1.

Toward this end, we examine the global error indicator Θ_k for element k which depends on h_k and p_k,

$$\Theta_k = \int_\Omega \theta_k(h, p)\, d\Omega \tag{19.2.30}$$

where $\theta_k(h, p)$ is the local error density. Thus, the total error indicator is expressed as

$$\Theta = \sum_k \Theta_k \tag{19.2.31}$$

Similarly, the total number of degrees of freedom is

$$N = \sum_k N_k = \int_\Omega n_k(h, p)\, d\,\Omega \tag{19.2.32}$$

where $n_k(h, p)$ denotes a degree of freedom density. Assume that the optimal mesh arises at $n = n_0$. Thus, the optimality condition can be achieved by constructing the Lagrange multiplier constraint

$$\lambda(n - n_o) = 0 \tag{19.2.33}$$

so that the functional

$$f = \theta(h, p) - \lambda(n - n_o) \tag{19.2.34}$$

achieves an optimality at

$$\delta f = \frac{\partial f}{\partial h}\delta h + \frac{\partial f}{\partial p}\delta p = 0 \tag{19.2.35}$$

Since δh and δp are arbitrary, we must have

$$\frac{\partial f}{\partial h} = \frac{\partial \theta}{\partial h} - \lambda\frac{\partial n}{\partial h} = 0 \tag{19.2.36}$$

$$\frac{\partial f}{\partial p} = \frac{\partial \theta}{\partial p} - \lambda\frac{\partial n}{\partial p} = 0 \tag{19.2.37}$$

These conditions lead to the optimal *hp* distribution,

$$\left.\frac{\partial \theta}{\partial n}\right|_{p=\text{constant}} = \lambda|_p \tag{19.2.38}$$

$$\left.\frac{\partial \theta}{\partial n}\right|_{h=\text{constant}} = \lambda|_h \tag{19.2.39}$$

The derivatives in (19.2.38) and (19.2.39) may be approximated by $\Delta\theta/\Delta n$, with $\Delta\theta$ denoting the change in error due to a change in number of degrees of freedom Δn. The process to reduce the error as much as possible would make the change in error per

change in number of degrees of freedom as large as possible. Thus, the larger of the two quantities,

$$\lambda|_p = \left.\frac{\Delta\theta}{\Delta n}\right|_p = \text{constant} \tag{19.2.40}$$

or

$$\lambda|_h = \left.\frac{\Delta\theta}{\Delta n}\right|_h = \text{constant} \tag{19.2.41}$$

should be used as the result of optimization.

Notice that to modify a trial mesh, one refines those elements with $|\Delta^+\theta_k|$ below λ and unrefines those for which $|\Delta^+\theta_k|$ is above λ. For optimality, we refine elements for which the anticipated decrease of the error per unit new degrees of freedom is the largest.

For two-dimensional problems, refinements are not restricted in one element. This is because the approximation inside two neighboring elements is affected by the *p*-enrichment and *h*-refinement causing subdivision of neighboring elements. However, it is possible to extrapolate the one dimensional strategy to perform refinements for which the anticipated decreases of the error per new degree of freedom are as large as possible.

It may be argued that raising p gives a larger decrease in error than subdividing the element for some problems, but the mesh is achieved when geometrically well graded toward singularity with low p. The general procedure for the *hp* process is as follows:

(1) Compute the anticipated degrees of errors for all elements in an initial mesh.
(2) Evaluate $\frac{\Delta\theta}{\Delta n}|_p$ and $\frac{\Delta\theta}{\Delta n}|_h$ for every element.
(3) Identify $(\frac{\Delta\theta}{\Delta n})_{\max} = A$.
(4) Identify those elements for which $\frac{\Delta\theta}{\Delta n} \geq \alpha A$ where α is a predetermined number for refinement.
(5) Perform refinements based on Steps (2) and (4) and solve the problem on the new mesh.
(6) Calculate the global error $\Theta = \sum_k \Theta_k$. If $\Theta \leq \beta$ where β is a predetermined error tolerance, then stop; otherwise go to (1).

In the process of *hp* refinements, it is frequently required that adjacent elements have larger or smaller degrees of polynomial approximations than the element under consideration. This will result in irregular elements with irregular nodes. In this case, the adjoining boundaries are dictated by the higher order approximations of either element.

Oden et al. [1995] reports numerical results for the incompressible flow Navier-Stokes solution using the three-step *hp* methods in which the following three steps are implemented:

(1) Estimate the error indicator (19.2.2) on the initial mesh
(2) Compute n_k in (19.2.32) to construct a second mesh
(3) Calculate the distribution of polynomial degrees p_k to construct a third mesh.

An application of the above procedure to a back-step channel problem [Oden et al., 1995] is presented in Figure 19.2.17 and Table 19.2.1. The geometry features of the

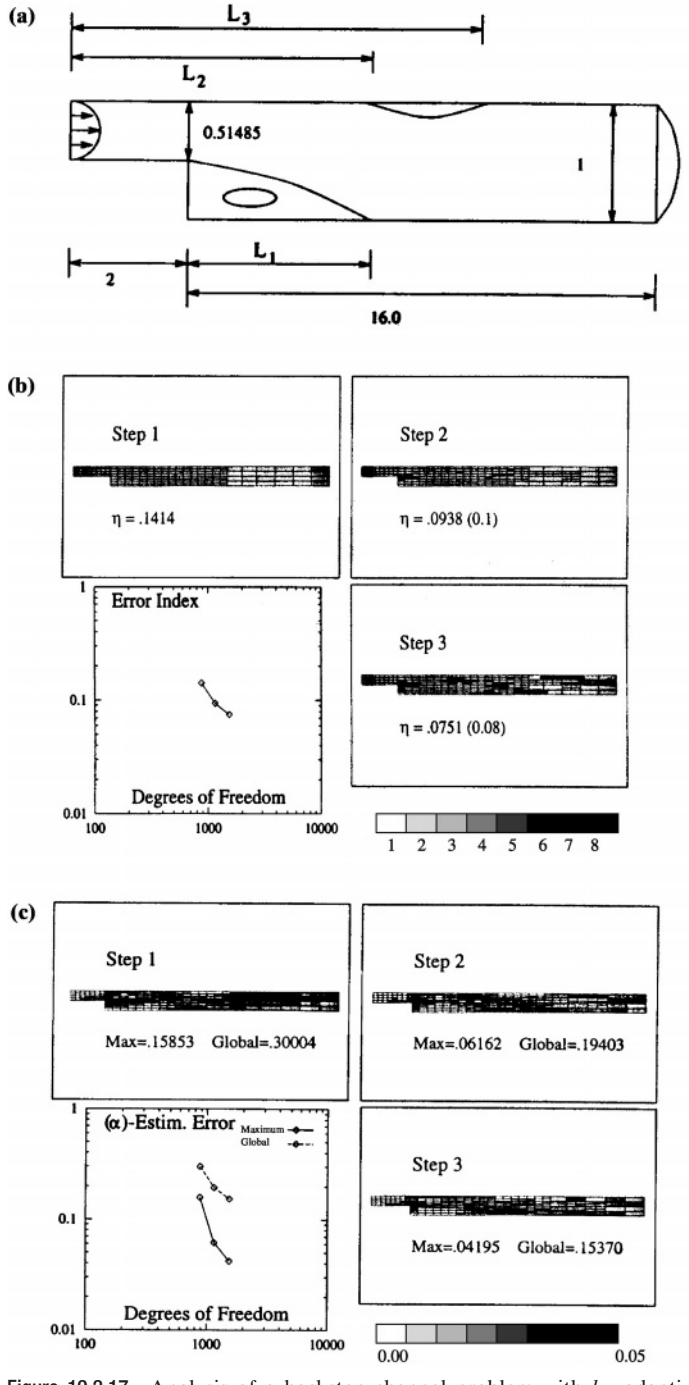


Figure 19.2.17 Analysis of a backstep channel problem with *hp* adaptive method (Rc = 300) [Oden et al., 1995]. **(a)** Geometry for the backstep problem. **(b)** Close-up view of the three adaptive meshes. **(c)** Equilibrated estimated error.

Table 19.2.1 CPU Time and Reattachment Length, Backstep Problem of Figure 19.2.17

(a) CPU Time

Mesh	CPU for the Solution (number of iterations)	CPU for the Error Estimates (equilibriated)	(0.5)
1	12246(21)	1283	866
2	3333(4)	2073	1171
3	9264(5)	3845	2787
Total	24843	7201	4824
	100%	28%	19%

(b) Comparison of Reattachment Lengths with Ghia et al. [1989]*

Reattachment Lengths	Reference Results*	Present Results
L_1	4.96	4.95
L_2	4.05	4.13
L_3	7.55	7.32

Sources: [Oden et al., 1995]

problem are defined in Figure 19.2.17a. An initial mesh of 877 scalar degrees of freedom and a quadratic interpolation are used. Close-up views of the three meshes and error index evolution and equilibrated estimated error are shown in Figures 19.2.17b,c. The elements are h-refined near the singularity and orders of $p = 4$ and $p = 3$ are assigned near this point. However, the adaptive strategy also leads to refinements and enrichments in other areas. In order to illustrate the cost of the adaptive strategy, Table 19.2.1a shows the CPU time used for each part of the calculation. The total number of iterations to reach the solution on each mesh (relative variation 10^{-9}) is also provided. Table 19.2.1b presents results in good agreement with the literature [Ghia et al., 1989].

Oden et al. [1998] further presented examples of hp methods applied to diffusion problems using a discontinuous Galerkin formulation. Here, arbitrary spectral approximations are constructed with different orders p in each element. The results of numerical experiments on h and p-convergence rates for representative two-dimensional problems suggest that the method is robust and capable of delivering exponential rates of convergence.

19.2.6 UNSTRUCTURED FINITE DIFFERENCE MESH REFINEMENTS

The control function methods and variational methods presented in Section 19.1 are suitable for structured grids only. After the adaptive process, the entire mesh still remains structured. In the mesh refinement methods, it is desirable that such restriction be removed even for the FDM formulation. We examine this possibility for FDM.

The simplest case of mesh refinement may be illustrated for finite difference formulations as demonstrated by Altas and Stephenson [1991]. Consider a square S given by

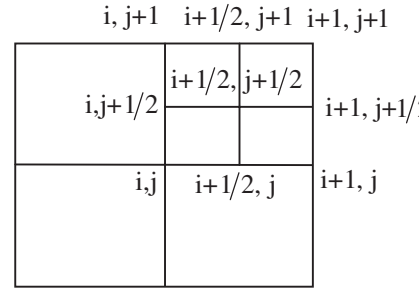


Figure 19.2.18 Comparison of errors between a square and subsquares.

(i, j), $(i+1, j)$, $(i+1, j+1)$, and $(i, j+1)$ and its subsquares, as shown in Figure 19.2.18. The computational error between the square and subsquares may be characterized as

$$e = \left| \left[\iint u(x, y) ds - e_1 \right] - \left[\iint u(x, y) ds - e_2 \right] \right| \tag{19.2.42}$$

where

$$e_1 = \frac{1}{4}(x_{i+1} - x_i)(y_{i+1} - y_i)[u(x_i, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j+1}) + u(x_{i+1}, y_{j+1})]$$

$$\begin{aligned} e_2 = \frac{1}{16}(x_{i+1} - x_i)(y_{i+1} - y_i)\{&u(x_i, y_j) + u(x_i, y_{j+1}) + u(x_{i+1}, y_{j+1}) + 2[u(x_{i+\frac{1}{2}}, y_j) \\ &+ u(x_{i+1}, y_{j+\frac{1}{2}}) + u(x_{i+\frac{1}{2}}, y_{j+1}) + u(x_i, y_{j+\frac{1}{2}})] + 4u(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})\} \end{aligned}$$

$$\begin{aligned} e &= |e_1 - e_2| \\ &= \frac{1}{16}(x_{i+1} - x_i)(y_{i+1} - y_i) \big| \{3[u(x_i, y_j) + u(x_{i+1}, y_j) + u(x_i, y_{j+1}) \\ &\quad + u(x_{i+1}, y_{j+1})] - 2[u(x_{i+\frac{1}{2}}, y_j) + u(x_{i+1}, y_{j+\frac{1}{2}}) + u(x_{i+\frac{1}{2}}, y_{j+1}) \\ &\quad + u(x_i, y_{j+\frac{1}{2}})] - 4u(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})\} \big| \end{aligned} \tag{19.2.43}$$

It can be shown using Taylor series expansions of the functions about the center point $(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}})$ of S that

$$\begin{aligned} e = \frac{1}{16}(x_{i+1} - x_i)(y_{i+1} - y_i) \big| &2(x_{i+1} - x_i)^2 u_{xx}(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}}) \\ &+ 2(y_{i+1} - y_i)^2 u_{yy}(x_{i+\frac{1}{2}}, y_{j+\frac{1}{2}}) + R \big| \end{aligned} \tag{19.2.44}$$

where R denote the remainder terms in Taylor expansions.

Here u is known only at vertices (Figure 19.2.18). Thus we construct a linear interpolation for side nodes and interior nodes. An adaptive mesh is created for all squares for which

$$e \geq E$$

where E is the user-defined tolerance.

(1) Start by using the subregions with a uniform mesh.
(2) Evaluate E using (19.2.44) on each subregion.

(3) Subdivide the regions with the quantity E larger than a given tolerance $\in$ into four equal subregions.
(4) On the new mesh points, either obtain a new approximate solution to the problem or use interpolated values of the previously obtained solution.
(5) Continue steps 2 through 4 until the largest value of E is less than $\in$.
(6) Solve the problem on the final mesh.

Some example problems using unstructured adaptive finite difference mesh refinements can be found in Altas and Stephenson [1991].

19.3 SUMMARY

Adaptive mesh methods were developed in structured grids using control functions and variational functions for FDM formulations. Obviously, in geometrical configurations not suitable for structured grids, control functions or variational functions are difficult to apply.

Unstructured adaptive methods have been extensively developed for FEM applications. Mesh refinement methods (h-methods) with error estimates and error indicators, mesh movement methods (r-methods), combined mesh refinement and mesh movement methods (hr-methods), mesh enrichment methods (p-methods), and combined mesh refinement and mesh enrichment methods (hp methods) were introduced in this chapter.

It is shown in Section 19.2.6 that adaptive unstructured mesh refinements can be performed by finite differences, although severely limited in utility and flexibility. Much greater efficiency can be provided with finite elements. For the last two decades, Oden and his co-workers and Babska and his co-workers have made significant contributions in FEM adaptive mesh methods. Developments of adaptive mesh methods in unstructured grids constitute one of the great achievements in the FEM research.

REFERENCES

Altas, I. and Stephenson, J. W. [1991]. A two-dimensional adaptive mesh generation method. *J. Comp. Phys.*, 94, 201–24.

Babuska, I. and Suri, M. [1990]. The p- and h-p versions of the finite element method. An overview. *Comp. Meth. Appl. Mech. Eng.*, 80, 5–26.

Babuska, I., Zienkiewicz, O. C., Gago, J., and Oliveira, E. R. A. (eds.) [1986]. *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*. Chichester: Wiley.

Brackbill, J. U. [1982]. Coordinate System Control: Adaptive Meshes, Numerical Geneneration, Proceedings of a Symposium on the Numerical Generation of Curvilinear Coordinate Systems and their Use in the Numerical Solution of Partial Differential Equations (J. F. Thompson, ed.), New York: Elsevier, 277–94.

Brackbill, J. U. and Saltzman, J. S. [1982]. Adaptive zoning for singular problems in two dimensions. *J. Comp. Phys.*, 46, 342.

Chung, T. J. [1996]. *Applied Continuum Mechanics*. New York: Cambridge University Press.

Devloo, P., Oden, J. T., and Pattani, P. [1988]. An adaptive h-p finite element method for complex compressible viscous flows. *Comp. Meth. Appl. Mech. Eng.*, 70, 203–35.

Dwyer, H. A., Smooke, D. Mitchell, and Kee, Robert, J. [1982]. Adaptive gridding for finite difference solutions to heat and mass transfer problems. In J. F. Thompson (ed.). *Numerical Grid Generation*, New York: North-Holland, 339.

Eiseman, P. R. [1985]. Alternating direction adaptive grid generation. *AIAA J.*, 23, 551–60.
———. [1987]. Adaptive grid generation. *Comp. Meth. Appl. Mech. Eng.*, 64, 321–76.
Ghia, K. N., Osswald, G. A., and Ghia, U. [1989]. Analysis of incompressible massively separated viscous flows using unsteady Navier-Stokes equations. *Int. J. Num. Meth. Fl.*, 9, 1025–50.
Gnoffo, P. A. [1980]. Complete supersonic flowfields over blunt bodies in a generalized orthogonal coordinate system. NASA TM 81784.
Heard, G. A. and Chung, T. J. [2000]. Numerical simulation of 3-D hypersonic flow using flowfield-dependent variation theory combined with an *h*-refinement adaptive mesh. Presented at FEF2000, The University of Texas/Austin.
Kim, H. J. and Thompson, Joe F. [1990]. Three-dimensional adaptive grid generation on a composite-block grid. *AIAA J.*, 28, no. 3, 470–77.
Lohner, R. and Baum, J. D. [1990]. Numerical simulation of shock interaction with complex geometry three-dimensional structures using a new adaptive *h*-refinement scheme on unstructured grids. 28th Aerospace Sciences Meeting, January 8–11, 1990, Reno, Nevada, AIAA 90-0700.
Nakamura, S. [1982]. Marching grid generation using parabolic partial differential equations. In J. F. Thompson (ed.). *Numerical Grid Generation*, New York: North-Holland, 775.
Oden, J. T. [1988]. Adaptive FEM in complex flow problems. In J. R.Whiteman (ed.). *The Mathematics of Finite Elements with Applications*, Vol. 6, London: Academic Press, Lt., 1–29.
———. [1989]. Progress in adaptive methods in computational fluid dynamics. In J. Flaherty, et al. (ed.). *Adaptive Methods for Partial Differential Equations*, Philadelphia: SIAM Publications.
Oden, J. T., Babuska, I., and Baumann, C. E. [1998]. A discontinuous *hp* finite element method for diffusion problems. *J. Comp. Phys.*, 146, 491–519.
Oden, J. T., Strouboulis, T., and Devloo, P. [1986]. Adaptive finite element methods for the analysis of inviscid compressible flow: I. Fast refinement/unrefinement and moving mesh methods for unstructured meshes. *Comp. Meth. Appl. Mech. Eng.*, 59, no. 3, 327–62.
Oden, J. T., Wu, W., and Legat, V. [1995]. An *hp* adaptive strategy for finite element approximations of the Navier-Stokes equations. *Int. J. Num. Meth. Fl.*, 20, 831–51.
Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C. [1987]. Adaptive remeshing for compressible flow computations. *J. Comp. Phys.*, 72, no. 2, 449–66.
Probert, J., Hassan, O., Peraire, J., and Morgan, K. [1991]. An adaptive finite element method for transient compressible flows. *Int. J. Num. Meth. Eng.*, 32, 1145–59.
Yoon, W. S. and Chung, T. J. [1991]. Liquid propellant combustion waves. Washington, D.C.: AIAA paper AIAA-91-2088.