8.1 INTRODUCTION

The first seven chapters of this text examine the essential aspects of finite-element semidiscrete approximation procedures applied to progressively more complete differential equation descriptions in fluid mechanics. In each instance, the basic Navier-Stokes equation system, governing completely general flow fields in three-dimensional configurations, was simplified by enforcing assumptions that restricted certain physical and/or geometric aspects. This chapter presents the finite-element algorithm construction for the complete three-dimensional Navier-Stokes equations governing compressible, turbulent flow fields.

The discipline pacing the development of algorithms for computational fluid dynamics (CFD) applications, for the general problem description, has been aerodynamics. The true "workhorse" of the CFD aerodynamics community over the past decade has been the MacCormack algorithm, originally published in 1969 (MacCormack, 1969)—and variations thereof. This is due principally to the ultimate simplicity of this explicit, predictor-corrector algorithm, and its proven track record for prediction of compressible, supersonic inviscid flow fields. In the split-operator construction, the programming requirements are elementary, and the resultant code runs quite economically. The basic theoretical formulation enjoys continuing refinement, including application to the viscous Navier-Stokes equations (MacCormack, 1977, 1981). In this instance, the added discretization refinement required to resolve wall layers "stiffens" the resulting discretized equation system. For an elementary explicit algorithm (recall Chap. 4), the integration absolute stability interval is bounded by a multiple of the largest eigenvalue of the discretized system matrix, thus yielding the requirement to time march using a very small step size.
For this reason and for its own merit, recent CFD research has turned to development of implicit Navier-Stokes algorithms. Simply stated, one trades the small time-step restriction for the requirement to solve matrix equation systems. As a result, principal attention has focused on matrix factorization procedures that reduce large sparse matrices to block-banded forms, to permit running solutions on present computer systems. The approximate-factorization, implicit finite-difference (AFFD) Navier-Stokes algorithms of Beam and Warming (1978) and Briley and McDonald (1977) exemplify the concept. The numerics are typically second-order accurate in space, first order in (pseudo-) time, and may employ both implicit and/or explicit artificial diffusion to control perceived instabilities. In the favored noniterative delta formulation, the matrix solution procedure constitutes acceptance of the first iterate of the Newton algorithm, using an approximation to the true matrix system Jacobian constructed upon addition of the identity matrix.

An integral ingredient of the AFFD algorithm construction has been definition and use of the “generalized coordinates” description. Basically, the divergence operator in the Navier-Stokes equation set is transformed into scalar components parallel to principal coordinates of a coordinate transformation “regularizing” the boundary $\partial R$ of the solution domain $R^n$ to the unit square (or cube, $n = 3$). Thompson and coworkers (Thames et al., 1977) pioneered the numerical transformation concept. Steger and Pulliam (1980) first reported results using the generalized coordinates AFFD algorithm concept for a two-dimensional cascade flow; topical results were recently summarized by Steger (1981).

In this chapter, we develop the finite-element algorithm for the problem class, that embraces as special cases the essential features of the AFFD algorithm. The basic finite-element theoretical statement is essentially identical to the dissipative, Galerkin weighted residuals algorithm derived in Chaps. 4 and 5, but generalized for a multidependent variable vector. The embedded dissipation parameter set $\beta_n$ is generalized and optimized for the problem class. The entire algorithm statement is then recast into the generalized coordinates framework, which facilitates construction of the tensor matrix product approximation to the Newton iteration algorithm Jacobian while retaining complete geometrical versatility. The remaining sections of this chapter quantize developmental aspects and document key numerical results regarding accuracy and convergence character.

8.2 THREE-DIMENSIONAL NAVIER–STOKES EQUATIONS

The partial differential equation set governing three-dimensional laminar and/or turbulent flow of a compressible, heat-conducting fluid is the familiar and very nonlinear Navier-Stokes system. In nondimensional divergence form, and using cartesian tensor summation notation, the equation set governing conservation of mass, momentum, and energy is

$$L(\rho) = \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j} (\rho u_j) = 0$$

(8.1)

$$L(p) = \frac{\partial (\rho u_j)}{\partial t} + \frac{\partial}{\partial x_j} (\rho u_j u_j + p \delta_{ij} - \sigma_{ij}) = 0$$

(8.2)

$$L(u_j) = \frac{\partial (\rho u_j u_j + p \delta_{ij} - \sigma_{ij})}{\partial t} + \frac{\partial}{\partial x_j} (u_j \rho u_j + u_j p - \sigma_{ij} u_j - \epsilon) = 0$$

(8.3)

In (8.1)-(8.3), $\rho$ is density, $u_j = m_j$ is the momentum vector, $p$ is pressure, and $\epsilon$ is mass specific total energy. The equation of state, for a polytropic gas $p = (\gamma - 1) \rho e$, is

$$L(p) = p - (\gamma - 1) (\rho e - \frac{1}{2} \rho u_j^2) = 0$$

(8.4)

The Stokes viscous stress tensor $\sigma_{ij}$, and heat flux vector $q_j$ in terms of the specific internal energy $e$, are defined as

$$\sigma_{ij} = \frac{\mu}{\Re} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2\mu}{3\Re} \frac{\partial u_k}{\partial x_k} \delta_{ij}$$

(8.5)

$$q_j = -\kappa \frac{\partial e}{\partial x_j}$$

(8.6)

$$e = e - \frac{1}{2} \mu u_j^2$$

(8.7)

where $\mu$ is the absolute viscosity, $\kappa$ is the coefficient of heat conductivity, $\delta_{ij}$ is the Kronecker delta, and $\Re$ is the reference Reynolds number.

The Euler equations, governing three-dimensional inviscid flows, are contained within (8.1)-(8.4), i.e., (8.5)-(8.6) vanish identically. The equation set (8.1)-(8.4) is also representative of the mass-weighted, time-averaged Navier-Stokes equations for a turbulent flow (Cebeci and Smith, 1974, Chap. 2). The dependent variables are interpreted as descriptors of the time-averaged mean flow, and $\bar{\sigma}_{ij}$ is generalized to include nonvanishing correlations of subgrid scale phenomena. In this instance, the total stress tensor becomes

$$\bar{\sigma}_{ij} = \bar{\sigma}_{ij} - \rho \bar{u}_i \bar{u}_j$$

(8.8)

where $-\rho \bar{u}_i \bar{u}_j$ is the dynamic Reynolds stress tensor, and $\bar{\sigma}_{ij}$ denotes the time-averaged form of (8.5). The exact differential equation system governing the dynamic Reynolds stress tensor is (Marvin, 1982)

$$L(\bar{u}_i \bar{u}_j) = \frac{\partial}{\partial t} (\bar{u}_i \bar{u}_j) + \frac{\partial}{\partial x_j} (\bar{u}_i \bar{u}_j \bar{u}_k) + \rho \bar{u}_j \frac{\partial \bar{u}_k}{\partial x_j} + \bar{u}_j \frac{\partial \bar{u}_i \bar{u}_k}{\partial x_j} + \frac{\partial}{\partial x_j} (\bar{u}_i \bar{u}_k \bar{u}_j)$$

$$+ \frac{\partial}{\partial x_k} (\bar{u}_i \bar{u}_k) + \frac{\partial}{\partial x_j} (\bar{u}_k \bar{u}_j) - \rho \left( \frac{\partial \bar{u}_k}{\partial x_j} + \frac{\partial \bar{u}_i}{\partial x_j} \right) - \frac{\partial}{\partial x_j} (\bar{u}_k \bar{u}_j)$$

$$- \frac{\partial}{\partial x_j} (\bar{u}_i \bar{u}_k) + \frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_k}{\partial x_j} = 0$$

(8.9)

In (8.9), the mass weighted time-averaged velocity is defined as

$$\bar{u}_i = \bar{\rho} \bar{u}_i$$

(8.10)
and the instantaneous velocity is assumed of the form
\[ u_i(x_j, t) = \bar{u}_i(x_j, t) + u'_i(x_j, t) \quad (8.11) \]

The trace of the stress tensor \(-\bar{\rho}u'_i u'_j\) defines the time-averaged turbulence kinetic energy \(\bar{k}\)
\[ \bar{k} = \frac{1}{2} \bar{\rho}u'_i u'_j \quad (8.12) \]

The corresponding transport equation for \(\bar{k}\) is established from (8.9) as (Marvin, 1982)
\[ L(\bar{k}) = \frac{\partial}{\partial t} (\bar{\rho} \bar{k}) + \frac{\partial}{\partial x_j} (\bar{p} \bar{u}_j \bar{k}) + \frac{\bar{\rho}u'_k}{\bar{\rho}} \frac{\partial u'_k}{\partial x_k} + \frac{\partial}{\partial x_j} (\bar{\rho}u'_i \bar{u}'_i) \]
\[ + \frac{\partial}{\partial x_k} \left( \bar{u}'_i \right) \left( \bar{u}'_i \right) + \frac{\partial}{\partial x_j} (\bar{u}'_j \bar{u}'_j) = 0 \quad (8.13) \]

For flow fields with associated Mach numbers though modest supersonic, the larger scale motion is statistically coupled to the thermal field almost exclusively through mean values of density, viscosity, and conductivity (Morkovin, 1964). Hence, for many flows of practical interest, it may be sufficient to assume
\[ \bar{\rho}u'_i u'_j \equiv \bar{\rho} \bar{u}'_i \bar{u}'_j \quad (8.14) \]

Assuming the validity of (8.14), the corresponding differential equation for the kinematic Reynolds stress tensor \(-\bar{u}'_i \bar{u}'_j\) is established from (8.9) and (8.14) in the form
\[ L(\bar{u}'_i \bar{u}'_j) = \frac{\partial}{\partial t} (\bar{u}'_i \bar{u}'_j) + \frac{\partial}{\partial x_k} (\bar{u}'_k \bar{u}'_i \bar{u}'_j) + \left( \bar{u}'_k \bar{u}'_i \frac{\partial \bar{u}'_k}{\partial x_k} + \bar{u}'_k \bar{u}'_j \frac{\partial \bar{u}'_k}{\partial x_k} \right) + \frac{\partial}{\partial x_k} \left( \bar{u}'_i \bar{u}'_j \right) \]
\[ + \frac{\bar{p}}{\bar{\rho}} \left( \frac{\partial \bar{u}'_i}{\partial x_j} + \frac{\partial \bar{u}'_i}{\partial x_k} \right) + \frac{\partial}{\partial x_k} \left[ \bar{u}'_i \frac{\partial \bar{u}'_j}{\partial x_k} - \nu \frac{\partial \bar{u}'_i}{\partial x_k} \right] + \frac{\bar{p}}{\bar{\rho}} \left( \delta_{ik} \bar{u}'_j + \delta_{jk} \bar{u}'_i \right) = 0 \quad (8.15) \]

where \(\bar{\nu} \equiv \bar{\nu}/\bar{\rho}\) is the time-averaged kinematic viscosity.

The fourth term in (8.15) defines the mechanism for viscous dissipation of the kinematic Reynolds stress tensor. The scalar, formed by the contraction of this term, is defined as the isotropic dissipation function \(\epsilon\)
\[ \frac{2}{3} \delta_{i\epsilon} = 2\bar{\rho} \bar{\nu}_i \bar{\nu}_j \frac{\partial \bar{u}'_i}{\partial x_k} \frac{\partial \bar{u}'_j}{\partial x_k} \quad (8.16) \]

The governing differential equation for the isotropic dissipation function \(\epsilon\) is modeled after (8.15) (Tennekes and Lumley, 1974) as
\[ L(\epsilon) = \frac{\partial \epsilon}{\partial t} + \frac{\partial}{\partial x_k} (\bar{u}_k \epsilon) + 2\bar{\nu}_i \bar{\nu}_j \frac{\partial \bar{u}'_i}{\partial x_k} \frac{\partial \bar{u}'_j}{\partial x_k} + 2 \left( \nu \frac{\partial^2 \bar{u}'_i}{\partial x_k^2} \right) \]
\[ + \frac{\partial}{\partial x_k} \left( \bar{\nu}'_k \frac{\partial \bar{u}'_i}{\partial x_k} + \nu \frac{\partial \bar{p}}{\partial x_k} \frac{\partial \bar{u}'_i}{\partial x_k} \right) = 0 \quad (8.17) \]

The companion transport equation for turbulence kinetic energy function \(k\),
\[ k \equiv \frac{1}{2} \bar{\rho}u'_i u'_i \quad (8.18) \]

is determined by contracting (8.15), yielding
\[ L(k) = \frac{\partial k}{\partial t} + \frac{\partial}{\partial x_j} (\bar{u}_j k) + \bar{u}'_i u'_i \frac{\partial k}{\partial x_j} + \epsilon + \left( \frac{\bar{p}}{\bar{\rho}} \frac{\partial \bar{u}'_i}{\partial x_j} \right) \]
\[ + \frac{\partial}{\partial x_j} \left( \bar{u}'_i u'_i - \nu \frac{\partial k}{\partial x_j} + \frac{\bar{p}u'_i}{\bar{\rho}} \right) = 0 \quad (8.19) \]

Even with the simplification of (8.14), (8.15)–(8.19) contain more unknowns than available equations, since the various third order correlations are undefined. Additional differential equations could be derived, but each set in turn would contain fourth and higher order correlations as unknowns. Therefore, the typical practice is to model the highest order correlations in the set (8.9)–(8.19). For example, using the procedures of Launder et al. (1975) (see also Hanjalic and Launder, 1972), and assuming validity of (8.14), the two-equation closure system modeling of (8.17) and (8.19) produces the familiar \(k-\epsilon\) system:
\[ L(k) = \frac{\partial k}{\partial t} + \frac{\partial}{\partial x_j} \left[ \bar{u}_j k + \left( C_k \bar{u}'_i \bar{u}'_j \frac{k}{\bar{\epsilon}} - \bar{\nu} \frac{\partial \bar{u}'_i}{\partial x_j} \right) \frac{\partial k}{\partial x_j} \right] + \frac{\partial}{\partial x_j} \left( \bar{u}'_i u'_i - \nu \frac{\partial k}{\partial x_j} + \frac{\bar{p}u'_i}{\bar{\rho}} \right) + \epsilon = 0 \quad (8.20) \]
\[ L(\epsilon) = \frac{\partial \epsilon}{\partial t} + \frac{\partial}{\partial x_j} \left[ \bar{u}_j \frac{\partial \epsilon}{\partial x_j} + \left( C_\epsilon \frac{\bar{u}'_i \bar{u}'_j}{k} \frac{\partial \epsilon}{\partial x_j} \right) + \frac{\bar{p}}{\bar{\rho}} \frac{\bar{u}'_i}{k} \frac{\partial \bar{u}'_i}{\partial x_j} + C_{\epsilon}^2 \frac{\bar{u}'_i}{k} \frac{\partial \bar{u}'_i}{\partial x_j} + C_{\epsilon}^2 \frac{\epsilon^3}{k} \right] = 0 \quad (8.21) \]

The standard values of the correlation coefficients \(C_k\) and \(C_\epsilon\), as determined by the analysis of Hanjalic and Launder (1972), are \(C_k = 1.0\), \(C_\epsilon = 1/1.3\), \(C_{\epsilon}^2 = 1.52\), and \(C_{\epsilon}^4 = 1.92\).

The solution to (8.20)–(8.21), coupled with appropriate boundary condition specifications, yields the distribution of the trace of the kinematic stress tensor \(-\bar{u}'_i \bar{u}'_j\) and \(\epsilon\). Therefore, an additional modeling is required to establish the relationship between \(k\) and \(\epsilon\) and \(-\bar{u}'_i \bar{u}'_j\). As presented in Chaps. 6 and 7, one approach which encompasses the bousinesq eddy viscosity model as a special case is the tensor constitutive equation
\[ \bar{u}'_i \bar{u}'_j = C_{\theta} \frac{\partial \bar{u}'_i}{\partial x_j} + \bar{C}_{\theta}^2 \frac{k^2}{\bar{\epsilon}} \left( \frac{\partial \bar{u}'_i}{\partial x_j} \right) + \left( \frac{\bar{C}_{\theta}^4}{\bar{\epsilon}} \frac{k^4}{\bar{\epsilon}} \right) \left( \frac{\partial \bar{u}'_i}{\partial x_j} \right) + \left( \frac{\bar{C}_{\theta}^2}{\bar{\epsilon}} \frac{k^2}{\bar{\epsilon}} \right) \left( \frac{\partial \bar{u}'_i}{\partial x_j} \right) + \left( \frac{\bar{C}_{\theta}^4}{\bar{\epsilon}} \frac{k^4}{\bar{\epsilon}} \right) \left( \frac{\partial \bar{u}'_i}{\partial x_j} \right) + \cdots \quad (8.22) \]

The various correlations \(C_{\theta}^\alpha\) were defined in Sec. 7.3 [see (7.22)–(7.24)].

The boundary conditions for the equation set (8.1)–(8.22) are a general mixture of Dirichlet and Neumann specifications. On an inflow boundary specification, \(\bar{p}, \bar{\rho}_u, \bar{u}, \bar{k}, \) and \(\epsilon\) are specified, and \(\bar{p}\) is determined from (8.4). For an inviscid streamline (Euler equations on a far-field boundary), the normal derivative of the scalar variables \(\bar{u}, \bar{\rho}_u, \bar{k}, \) and \(\epsilon\) vanish identically, and the normal derivative of \(\bar{p}\) is constrained (adiabatic or...
cooled wall). At an outflow boundary, typically, \( p \) is specified and the normal derivative of the remaining dependent variables is constrained.

### Problems
1. Establish (8.13) from the trace of (8.9).
2. Derive (8.15) from (8.9) using (8.14).
3. Establish (8.19) using (8.15) and (8.18).

#### 8.3 Finite-Element Solution Algorithm

Equations (8.1)–(8.22) constitute a general definition of the three-dimensional Navier-Stokes equations. In the limit \( \text{Re} \to \infty \), (8.5) vanishes identically; assuming that the flow is nonturbulent, (8.1)–(8.4) reduce to the Euler equations governing an inviscid rotational flow. Specifying \( m_i \equiv \rho u_i \), \( g \equiv \rho e \), define the vector-valued dependent variable set as

\[
q_0(x_i, t) \equiv (q_0) \equiv (\rho, m_i, g, p, a_{ij}, q_j, k, e)^T
\]  

Equations (8.1)–(8.3), and (8.20)–(8.21) each constitute an initial-value description of the general form

\[
L(q_0) = \frac{\partial q_0}{\partial t} + \frac{\partial}{\partial x_i} (u_j q_0 + f_{q0}) + s_q = 0
\]  

In (8.24), \( f_{q0} \) and \( s_q \) are specific nonlinear functions of their argument, determined by inspection. The remaining algebraic and partial differential equations [cf. (8.4)–(8.7) and (8.22)] are each of the form

\[
L(q_0) = q_0 + f_{q0} = 0
\]  

where the \( f_{q0} \) are also determined by inspection. Note in particular that (8.25) is simply a special form of (8.24).

The \( n \)-dimensional partial differential equation system (8.24)–(8.25) is defined on the euclidean space \( \mathbb{R}^n \), spanned by the coordinate system with scalar components \( x_j \), \( 1 \leq j \leq n \). The solution domain \( \Omega \) is defined as the product of \( \mathbb{R}^n \) and \( t \), for all elements of \( x \) belonging to \( \mathbb{R}^n \) and all elements of \( t \) belonging to the open interval measured from \( t_0 \), that is,

\[
\Omega = \mathbb{R}^n \times t = \{(x, t): x \in \mathbb{R}^n \text{ and } t \in (t_0, t)\}
\]  

The boundary \( \partial \Omega \) of the solution domain is the product of the boundary \( \partial \mathbb{R} \) of \( \mathbb{R}^n \), spanned by \( x \), and \( t \), i.e., \( \partial \Omega = \partial \mathbb{R} \times t \). Thereupon, the generalized form for a differential boundary constraint is

\[
l(q_0) = \alpha^2 q_0 + \alpha^2 \frac{\partial}{\partial x_i} (q_0 \hat{n}_i) + s^2 = 0
\]  

where the \( \alpha_i^2 \) are specified coefficients and \( \hat{n}_i \) is the outward pointing unit normal vector. Finally, an initial distribution for \( q_0 \) on \( \Omega_0 = \mathbb{R}^n \times t_0 \) is required; hence,

\[
q_0(x, t_0) = q_0^0(x)
\]

The finite-element algorithm for (8.24)–(8.27) is an extension of the original statement given in Secs. 4.10–4.11. The semidiscrete approximation \( q_0^e(x, t) \) to the unknown exact solution \( q_0(x, t) \) is constructed from members of a convenient finite-dimensional subspace of \( H^1_0(\Omega) \), the Hilbert space of all functions possessing square integrable first derivatives and satisfying the boundary conditions. The typical practice is to employ polynomials truncated to degree \( k \), and defined on disjoint interior subdomains \( \mathcal{R}_e \), the union of which forms the discretization of \( \mathbb{R}^n \equiv \cup \mathcal{R}_e \). Hence,

\[
q_0^e(x, t) = \sum_{e=1}^{M} q_0^e(x, t)
\]

and

\[
q_0^e(x, t) = \langle N_k(x) \rangle^T (Q(t))_e
\]

In (8.30), the semidiscrete free index \( \langle \cdot \rangle \) denotes \( q_0^e \) evaluated at the nodal coordinates of the discretization \( \cup \mathcal{R}_e \) at any time \( t \). The sub- or superscript \( e \) denotes pertaining to the \( e \)-th finite-element domain, \( \Omega_e = \mathcal{R}_e \times t \). The elements of the row matrix \( (N_k(x))^T \) are polynomials written on \( x_j \), \( 1 \leq j \leq n \), complete to degree \( k \), as discussed in Chap. 3.

The finite-element algorithm statement is the familiar form

\[
\int_{\mathbb{R}^n} \{N_k\} L(q_0^e) \, dx + \beta_1 \int_{\Omega} \{N_k\} \{Q(t)\} \, dx + \beta_2 \int_{\mathbb{R}^n} \{N_k\} \nabla L(q_0^e) \, dx = 0
\]

Upon definition of \( k \) in (8.30), (8.31) is a mixed system of ordinary differential equations written on \( t \), and algebraic equations of the form

\[
\{C\} \{Q(t)\} + \{U\} \{Q(t)\} + \{F(U)\} \{Q(t)\} + \{S\} = 0
\]

As discussed previously, the form of (8.32) suggests use of an implicit integration algorithm. The \( \theta \)-implicit finite-difference algorithm, where \( \theta = \frac{1}{2} \) yields the trapezoidal rule, yields

\[
\{F(T)\} = \{Q(t)\}_{t+1} - \{Q(t)\}_t - \Delta t [\theta \{Q(t)\}_{t+1} + (1 - \theta) \{Q(t)\}_t] = 0
\]

and (8.32) provides the definition of the derivatives \( \{Q(t)'\} \). Upon substitution, and proceeding through the algebra, (8.33) becomes a nonlinear algebraic equation system written on \( \{Q(t)\}_{t+1} \). The Newton iteration algorithm solution statement is

\[
\{J(FT)\}^{p+1}_{t+1} \{\delta Q(t)\}^{p+1}_{t+1} = -\{F(T)\}^{p+1}_t
\]

The dependent variable is the iteration vector \( \{\delta Q(t)\} \), yielding the solution in the conventional manner,

\[
\{Q(t)\}^{p+1}_{t+1} = \{Q(t)\}^{p+1}_t + \{\delta Q(t)\}^{p+1}_{t+1}
\]
The Jacobian of the Newton algorithm is defined as

\[ (K(F)) = \frac{\partial (F)}{\partial (Q)} \]  

(8.36)

8.4 GENERALIZED COORDINATES FORMULATION

A principal requirement in three-dimensional problem descriptions is to accurately define domain boundary geometries \( \partial \Omega \) that are nonregular and perhaps nonsmooth. The term generalized coordinates has gained acceptance in describing algorithm statements appropriate for use with regularizing boundary-fitted coordinate transformations, as discussed in Sec. 8.1. Many procedures are available to construct such transformations, including numerical solution of elliptic and hyperbolic partial differential equations, algebraic methods, and finite-element isoparametric interpolation (NASA, 1980).

The output of any of these procedures constitutes definition of the coordinate triples (pairs) on \( R^3 (R^2) \) that define the intersections of \( \partial R^3 \) on \( \Omega \), called nodes. The generalized coordinates description requirement is to construct the local coordinate transformation, using these data, that maps \( x^i \in R^3 \) to a regularized domain spanned by an orthonormal coordinate system \( \eta_i \). Figure 8.1 illustrates the concept, where \((\cdot)^T\) and \(x\) depict the nodal coordinates in both spaces. The coordinate transformation is

\[ x_i = x_i(\eta) \]  

(8.37)

The explicit form for (8.37) on \( R^3 \) is constructed (recall Sec. 3.6) as

\[ x_i = (N^T(\eta)) (XI)_e \]  

\[ x_i \in R^3 \]  

(8.38)

![Diagram](a)

![Diagram](b)

Figure 8.1 Biquadratic cardinal basis coordinate transformation. (a) Two-dimensional domain; (b) three-dimensional domain.

The elements of the column matrix \((XI)_e\) are the coordinates of the nodes of \( R^3 \), \( 1 \leq i, j \leq n \), hence \( \Omega R^3 \), the global discretization. For convenience, recall on \( R^2 \),

\[ (N^T(\eta)) = \frac{1}{4} \begin{pmatrix} (1 - \eta_1)(1 - \eta_2) \\ (1 + \eta_1)(1 - \eta_2) \\ (1 + \eta_1)(1 + \eta_2) \\ (1 - \eta_1)(1 + \eta_2) \end{pmatrix} \]  

(8.39)

For a discretization on \( R^2 \) using curved-sided serendipity quadrilaterals,

\[ (N^T(\eta)) = \frac{1}{4} \begin{pmatrix} (1 - \eta_1)(1 - \eta_2)(1 - \eta_1 - \eta_2 - 1) \\ (1 + \eta_1)(1 - \eta_2)(\eta_1 - \eta_2 - 1) \\ (1 + \eta_1)(1 + \eta_2)(\eta_1 + \eta_2 - 1) \\ (1 - \eta_1)(1 + \eta_2)(1 - \eta_1 + \eta_2 - 1) \end{pmatrix} \]  

(8.40)

For discretization of \( R^3 \) employing planar-faced hexahedra,

\[ (N^T(\eta)) = \frac{1}{6} \begin{pmatrix} (1 - \eta_1)(1 - \eta_2)(1 - \eta_3) \\ (1 + \eta_1)(1 - \eta_2)(1 - \eta_3) \\ (1 - \eta_1)(1 + \eta_2)(1 - \eta_3) \\ (1 + \eta_1)(1 + \eta_2)(1 - \eta_3) \\ (1 - \eta_1)(1 - \eta_2)(1 + \eta_3) \\ (1 + \eta_1)(1 - \eta_2)(1 + \eta_3) \\ (1 + \eta_1)(1 + \eta_2)(1 + \eta_3) \\ (1 - \eta_1)(1 + \eta_2)(1 + \eta_3) \end{pmatrix} \]  

(8.41)

The basic issue for the generalized coordinates form of the finite element algorithm is transformation of the divergence operator in (8.24) i.e.,

\[ \frac{\partial}{\partial x_j} = \frac{\partial \eta_i}{\partial x_j} \frac{\partial}{\partial \eta_i} \]  

(8.42)

Recalling Sec. 3.6, the elements of the inverse jacobian \( J^{-1} \) are

\[ \left[ \frac{\partial \eta_i}{\partial x_j} \right] = J^{-1} = \frac{1}{\det J} \left[ \text{transformed cofactor of } J \right] \]  

(8.43)
where \( J \) is the Jacobian of the forward transformation (8.38). The differential element for (8.31) is

\[
dx = \det[J] \, d\eta \tag{8.44}
\]

To illustrate the details, consider (8.31) written for the momentum equation (8.2). Using a Green-Gauss form of the divergence theorem, the first constraint term becomes

\[
\int_{R}^{n} (N) L(\rho u_{r}^{h}) \, dx = \int_{R}^{n} (N) \frac{\partial u_{r}^{h}}{\partial t} \det[J] \, d\eta \\
+ \int_{\partial R}^{n} \left( (N)(u_{r}^{h} \rho u_{r}^{h} + p^{h} \sigma_{j}) \right) \hat{n} \det[J] \, d\sigma \\
- \int_{R}^{n} \frac{\partial (N)}{\partial \eta_{j}} \left( \frac{\partial \eta_{j}}{\partial x_{j}} \right) \left( u_{r}^{h} \rho u_{r}^{h} + p^{h} \sigma_{j} - a_{j} \right) \det[J] \, d\eta
\tag{8.45}
\]

Define the contravariant components of the convection velocity as

\[
u_{r}^{h} = \det[J] \frac{\partial \eta_{k}}{\partial x_{j}} u^{h}_{j} \tag{8.46}
\]

with scalar components parallel to the \( \eta_{k} \) coordinate system. Using the algebraic transformation (8.38) and (8.43), \( \det[J] \) cancels yielding

\[
u_{r}^{h} = \text{Cof.} J \nu \tag{8.47}
\]

where \( \text{Cof.} J \) is the transformed cofactor matrix of \( J \). Using (8.30) to interpolate the nodal distribution of \( \nu_{r}^{h} \) on \( R_{e}^{n} \), (8.45) becomes

\[
\int_{R}^{n} (N) L(\rho u_{r}^{h}) \, dx = S_{e} \left[ \int_{R_{e}^{n}} (DET)^{T} (N)(N)^{T}(RHOUI)^{T}_{e} \, d\eta \\
- \int_{R_{e}^{n}} [UBARK]^{T} (N) \frac{\partial}{\partial \eta_{k}} (N)(N)^{T}(RHOUI)_{e} \, d\eta \\
- \int_{R_{e}^{n}} [ETAKI]^{T}_{e} (N) \frac{\partial}{\partial \eta_{k}} (N)(N)^{T}(P)_{e} \, d\eta \\
+ \int_{R_{e}^{n}} [ETAKI]^{T} (N) \frac{\partial}{\partial \eta_{k}} (N)(N)^{T}(SIGI)_{e} \, d\eta \\
+ \int_{\partial R_{e} \cap \partial R} [UBARK]^{T} (N)(N)(N)^{T}(RHOUI)_{e} \\
+ (DET)^{T}_{e} (N)(N)(N)^{T}(P)_{e} \delta_{jk} - (SIGI)_{e}) \hat{n} \, d\sigma \right] \tag{8.48}
\]

where \( S_{e} \) is the matrix operator projecting element contributions to the corresponding global matrices. The determinant of \( J \) is interpolated on the element \( R_{e}^{n} \) using \( (N)_{e}^{h} \) and the nodal values. Similarly, \( \frac{\partial \eta_{k}}{\partial x_{j}} \) is interpolated as \( (ETAKI)_{e}^{T} (N)_{e}^{h} \). Since the \( \epsilon \)-subscripted terms in (8.48) are independent of \( \eta_{k} \), upon extraction from the integrand, the finite-element algorithm statement, expressed in terms of the standard hypermatrix definitions, is

\[
\int_{R}^{n} (N) L(\rho u_{r}^{h}) \, dx = S_{e} \left[ (DET)^{T}_{e} (M3000) (RHOUI)^{T}_{e} \\
- [UBARK]^{T}_{e} (M30KO) (RHOUI)_{e} \\
- [ETAKI]^{T}_{e} (M30KO) (P)_{e} \\
+ [ETAKI]^{T}_{e} (M30KO) (SIGI)_{e} \\
+ \hat{n} \frac{1}{2} [UBARK]^{T}_{e} (N3000) (RHOUI) \\
+ (DET)^{T}_{e} (N3000)(P)_{e} \delta_{jk} - (DET)^{T}_{e} (N3000)(SIGI)_{e}] \right] \tag{8.49}
\]

In (8.49), the indices \( K \) and \( L \) obey the tensor summation rule, \( I \) is the free index for \( \rho u_{r}^{h} \), and \( (M30KO) \) is the element-independent hypermatrix equivalent of \( \frac{\partial}{\partial \eta_{j}} \) integrated over \( R_{e}^{n} \). The matrix \( (N3000) \) results from interpolation integrals on the element-boundary intersection \( \partial R_{e} \cap \partial R \) with outward normal \( \hat{n}_{j} \). For the coordinate transformation, \( (DET)_{e} \) is the nodal distributions of \( J^{-1} \) on \( R_{e}^{n} \). Within the generalized coordinates framework, therefore, the grid and metric data required for a numerical simulation are the nodal distributions of \( J^{-1} = \frac{\partial \eta_{j}}{\partial x_{j}} \), and \( \det[J] = \det[\partial x_{j}/\partial \eta_{j}] \). These data are strictly a function of the nodal coordinate distribution of the discretization \( UR_{e}^{n} \), and may be generated using any coordinate transformation procedure.

As developed in Sec. 8.6, \( \beta_{e} = (DET)_{e} \nu_{e}^{h} \) is the functional form for the dissipation term in the finite-element algorithm statement (8.31). With this identification, the specific matrix forms of the algorithm statement \( (FI) = 0 \), (8.33) can be expressed. The global matrix statement is always obtained using \( S_{e} \), on the sequence of elemental definitions, e.g.,

\[
(FI) = S_{e} [(FI)_{e}] \tag{8.50}
\]

The elemental expression \( (FI)_{e} \) is computed using the master hypermatrices \( (M \cdots) \) contracted with element-dependent matrices. Denote the members of the discrete dependent variable set as \( (Q)^{T} = (R, M, G, P, S, J, Q(J), K, E) \). The matrix algorithm statements \( (FI)_{e} \), for arbitrary basis \( (N)_{e}^{h} \) and dimension \( n \), are:

\[
(FI)_{e} = [(DET)^{T}(M3000) + \nu_{h} \frac{1}{2} (ETAKI)^{T}(M40K00)(DET)](R)_{j+1}^{T} \\
+ \frac{\Delta t}{2} [-(ETAKI)^{T}(M30K00)(M)] \\
+ \nu_{h} \frac{1}{2} (ETAKI)^{T}(M40K010)(UBARL)(R)]_{j+1, j} \tag{8.51}
\]
\[ (F_{M})_e = [(\eta - P)T \mathbf{M}_{000} + \nu_{\eta}^T (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta - P)] \mathbf{M} ]_{j+1}, \]
\[ + \frac{\Delta t}{2} - (\mathbf{U} \mathbf{B} - \mathbf{R}) \mathbf{M}_{000} (\eta) - (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta) \{P\} ] \mathbf{M} ]_{j+1}, \]
\[ + (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta) \{G\} ] \mathbf{M} ]_{j+1}, \]
\[ (FG)_{e} = [(\eta - P)T \mathbf{M}_{000} + \nu_{\eta}^T (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta - P)] \mathbf{M} ]_{j+1}, \]
\[ + \frac{\Delta t}{2} - (\mathbf{U} \mathbf{B} - \mathbf{R}) \mathbf{M}_{000} (\eta) + (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta) \{G\} ] \mathbf{M} ]_{j+1}, \]
\[ (\bar{F}G)_{e} = [(\eta - P)T \mathbf{M}_{000} + \nu_{\eta}^T (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta - P)] \mathbf{M} ]_{j+1}, \]
\[ + \frac{\Delta t}{2} - (\mathbf{U} \mathbf{B} - \mathbf{R}) \mathbf{M}_{000} (\eta) + (\mathbf{E} - \eta) \mathbf{M}_{000} (\eta) \{G\} ] \mathbf{M} ]_{j+1}, \]
\[ (FP)_{e} = \frac{2}{(\eta - P)T \mathbf{M}_{000} \{P\} - (\eta - 1)} [(\eta - P)T \mathbf{M}_{000} \{G\} ] + \frac{1}{2} \{D\}_{j+1} = \mathbf{M}_{000} \{U\} \{M\} \] \]
\[ \mathbf{M} ]_{j+1}, \]

\[ (FSIJ)_{e} = (\eta - P)T \mathbf{M}_{000} \{S\} + \frac{\mu}{Re} [(\mathbf{E} - \eta) \mathbf{M}_{000} \{S\} ] \mathbf{M} ]_{j+1}, \]
\[ + (\mathbf{E} - \eta) \mathbf{M}_{000} \{S\} - \frac{3}{2} \frac{\delta_{ij}}{\mathbf{M}_{000}} (\mathbf{M} \{S\} \{S\} ) \mathbf{M} ]_{j+1}, \]
\[ (FQ)_{e} = \frac{2}{(\eta - P)T \mathbf{M}_{000} \{Q\} - \kappa} [(\eta - P)T \mathbf{M}_{000} \{G\} ] + \frac{1}{2} \{D\}_{j+1} = \mathbf{M}_{000} \{U\} \{M\} \] \]
\[ \mathbf{M} ]_{j+1}, \]

A few comments on notational structure for (8.51)-(8.58) are appropriate. For clarity, the subscript \(e\) has been deleted from each equation. The elemental hypermatrices \([M]\) for an n-dimensional problem description, are evaluated once and for all by integrals over a master finite-element domain \(R^n\). For a one-dimensional domain \(R^1\), that is, \([A]\), all matrices through \([A A A A]\) are listed in Appendix B.1, for both the linear (\(k = 1\)) and quadratic (\(k = 2\)) cardinal basis \((N_k)\). For two- and three-dimensional domains \(R^n\), that is, \([B B B B]\) and \([C C C C]\), all element matrices through \([B B B B]\) and \([C C C C]\) are listed in Appendix B.2 and B.3, respectively, for the bilinear tensor product cardinal basis \((N^n)\). The discrete indices \(J, K, L\), occurring in both matrix and variable (FORTRAN) names, are tensor summation indices with range \(1 \leq (J, K, L) \leq n\). The multipole coefficient \(\beta_2\) is expressed in terms of cartesian scalar components \(\beta_2\), with distinct values for each dependent variable.

The arrays \((D_{\eta})_e\) and \((\mathbf{E} - \eta)\) contain nodal values of the determinant \(J\) and elements of \(\eta_{i,j}\). The nodal contravariant components of convection velocity \(\mathbf{u}\) \((8.47)\) are denoted \((\mathbf{U} \mathbf{B} - \mathbf{R})\), and \((\mathbf{U} \mathbf{B} - \mathbf{R})\) contains absolute values. The elements of \(\{S\}\) are nodal values of the stress tensor computed in principal coordinates in terms of \(u_j = m_{ij,p}\). Equation (8.55) is appropriate only for the laminar-flow Stokes stress tensor definition (8.5), and \(\delta_{ij}\) is the Kronecker delta. The additional terms resulting from inclusion of the Reynolds stress constitutive equation (8.22), are readily evaluated using the illustrated expansions. In (8.56), \(\{G\}\) denotes nodal values of \(e \equiv g_{i,j,p}\), while \(\{U\}\) contains nodal values of the specific kinetic energy \(\frac{1}{2} u_i u_j\). For (8.57)-(8.58), \(\{K\}\) and \(\{D\}\) contain as elements the nodal values of the "effective diffusion" coefficient tensors, \(C_{ik} \nu_{i,j,k}\) and \(C_{ik} \nu_{i,j,k}\), respectively. Further, in (8.58), the elements of \(\{S\}\) and \(\{S\}\) are nodal values of \(C_{ik} \nu_{i,j,k}\) and \(C_{ik} \nu_{i,j,k}\), respectively. In all equations, \(\chi_{j+1}\) denotes \(\chi_{j+1} - \chi_j\). The notation \(\chi_{j+1}\) defines evaluation of the argument at \(t_{j+1}\), and at \(t_j\), followed by addition after multiplication by \(\theta\) and \((1 - \theta)\), see (8.33). Equations (8.51)-(8.58) are completed by addition of the appropriate boundary terms, recall (8.47)-(8.49).

Problems
1 Derive (8.51).
2 Derive (8.52).
3 Derive (8.53)-(8.54).
4 Derive (8.55)-(8.56).
5 Derive (8.57)-(8.58).

8.5 TENSOR MATRIX PRODUCT JACOBIAN

For efficiency, a suitable approximation to the Newton algorithm Jacobian (8.34) is required (recall Chap. 4). The matrix product construction (Hallmass, 1958) can be achieved using the tensor product cardinal basis function set \(N^k(n)\) spanning quadrilateral and hexahedra element domains \(R^n\) and \(R^n\), respectively (Sec. 4.10). The Jacobian matrix \([\mathbf{J}(\mathbf{F})]\) \((8.34)\) is replaced by the tensor (outer) product, defined as

\[ [\mathbf{J}(\mathbf{F})] = [\mathbf{J}_1] \otimes [\mathbf{J}_2] \otimes [\mathbf{J}_3] \] \]

Each component \([\mathbf{J}_1]\) is constructed from its definition (8.36), assuming interpolation and differentiation are one-dimensional. Using (8.59), the Newton iteration statement (8.34) becomes replaced as

\[ [\mathbf{J}_1] \otimes [\mathbf{J}_2] \otimes [\mathbf{J}_3] [\mathbf{F}]_{j+1} = [\mathbf{F}]_{j+1} \] \]
Define
\[ [J_1] \otimes [J_2] \{\partial Q\}^{p+1}_{j+1} = \{P1\}^{p+1}_{j+1} [J_2] \{\partial Q\}^{p+1}_{j+1} = \{P2\}^{p+1}_{j+1} \]
\[ [J_3] \{\partial Q\}^{p+1}_{j+1} = \{F1\}^{p+1}_{j+1} \] \hspace{1cm} (8.61)

Then, the operation defined in (8.60) is replaced by the sequence
\[ [J_1] \{P1\}^{p+1}_{j+1} = -\{F1\}^{p+1}_{j+1} \]
\[ [J_2] \{P2\}^{p+1}_{j+1} = \{P1\}^{p+1}_{j+1} \]
\[ [J_3] \{\partial Q\}^{p+1}_{j+1} = \{P2\}^{p+1}_{j+1} \] \hspace{1cm} (8.62)

Obviously, other permutations of the index structure for \([J_3]\) could be utilized. The key aspect is the replacement of the very large (albeit sparse) Jacobian matrix \([J]\), with \(\alpha\)-block-structured matrices \([J_\eta]\). The principal attributes are several orders of magnitude reduction in central memory requirements for the Jacobian, and significantly reduced CPU for the LU decomposition and back substitution solution steps. The principal detraction is assured degradation of the quadratic convergence rate for the Newton iteration. This procedure in no way affects the formation of \((F1)\), (8.33), wherein lies the accuracy features intrinsic to the finite-element algorithm statement. Compromises in the evaluation of \((F1)\) will invariably produce inferior results for the Navier-Stokes equations.

The construction of the \([J_\eta]\) was introduced in Chap. 4 for a scalar equation. The procedures are directly extended to the present case. For example, the term in the Jacobian corresponding to the initial-value term in the matrix statements (8.51)-(8.58) is
\[ \frac{\partial \{F1\}}{\partial \{Q\}} = S_e \{\{DET\}_e^T [M3000] \delta_{J\eta} \} = [\{QQ\}] \] \hspace{1cm} (8.63)

where \(\delta_{J\eta}\) is the discrete index Kronecker delta. By definition, (8.63) is formed using \(S_e\) operating on the elemental definition
\[ [\{QQ\}]_e = \{\{DET\}_e^T [M3000] \} = \int_{R_3^1} \{\{DET\}_e^T [N_\eta(n)] \{N_\eta(n)\} \{N_\eta(n)\}^T d\eta \] \hspace{1cm} (8.64)

Assuming for exposition, \(k = 1, n = 2,\) and \(x = \eta (\) the identity transformation), and recalling the definition \(M = B\) for \(n = 2\), (8.64) becomes
\[ \{\{DET\}_e^T [B3000] \} = \Delta_e [B200] = \int_{R_2^1} \{N_1(x) \{N_1(x)\}^T dx \] \hspace{1cm} (8.65)

Assuming the rectangular element domain \(R_2^1\) described by measures \(l\) and \(\omega\), the evaluation of (8.65) yields
\[ [\{QQ\}]_e = \Delta_e [B200] = l \omega [B200] = \frac{l \omega}{36} \begin{bmatrix} 4 & 2 & 1 & 2 \\ 4 & 2 & 1 \\ 4 & 2 \\ \text{(sym)} & 4 \end{bmatrix} \] \hspace{1cm} (8.66)

The tensor product approximation to (8.63) involves evaluation of (8.65) on a one-dimensional domain. Recalling \(M = A\) for \(n = 1\),
\[ [\{QQ\}]_e = \Delta_e [A200] = \int_{R_2^1} \{N_1(x) \{N_1(x)\} \{N_1(x)\}^T dx \] \hspace{1cm} (8.67)

and
\[ [\{QQ\}]_e = l \omega \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \] \hspace{1cm} (8.68)

assuming \(\Delta_1 = l\) and \(\Delta_2 = \omega\). Accounting for entry locations in \([\{QQ\}]\), (8.63), and using the assembly operator \(S_e\), it is readily verified that
\[ [\{QQ\}] = S_e \{[\{QQ\}]_e \otimes [\{QQ\}]_e \} \] \hspace{1cm} (8.69)

By direct extension then, the tensor matrix product approximation of (8.63) on \(R^3\) is
\[ S_e \{\{DET\}_e^T [M3000] \} = S_e \{\{DET1\}_e^T [A3000] \otimes \{DET2\}_e^T [A3000] \}
\[ \otimes \{DET3\}_e^T [A3000] \} \] \hspace{1cm} (8.70)

where the tensor index \(1 \leq K \leq 3\) on \([\{DET\}]\) denotes the scalar component of the multidimensional measure parallel to \(\eta_k\).

The second term in the typical Jacobian [see (8.51)-(8.52)] is the matrix
\[ S_e \{\{DET\}_e^T [M3000] \} \]

where \(\nu_{J\eta}\) results from definition of the explicit form for \(p_k,\) (8.31), and \(\eta\) denotes the specific dependent variable. The indices \(K, J\) are discrete tensor summation indices with range \(1 \leq (K, J) \leq n.\) \([M3000]\) is a hypermatrix of degree two, i.e., it possesses elements which are themselves square matrices, which represents the integration over \(R_2^1\) of products of four cardinal basis sets \([N_\eta(n)]\), one of which is differentiated into scalar components parallel to \(\eta_k\). The tensor product approximation is
\[ S_e \{\{DET\}_e^T [M3000] \} = S_e \{\{DET\}_e^T [A40100] \otimes \{DET\}_e \}
\[ \otimes \{DET2\}_e^T [A40100] \otimes \{DET3\}_e^T [A40100] \otimes \{DET\}_e \} \] \hspace{1cm} (8.71)

where \(J\) is a discrete summation index with range \(1 \leq J \leq n\), to be contracted with scalar components of \(\nu_{J\eta}\).

Equations (8.70)-(8.71) illustrate the construction of tensor matrix product components of \([J]\). Each one-dimensional component \([J_\eta]\) for any term is identical in appearance; note that \([A40100]\) appears in each term in (8.71). Therefore, the tensor product Jacobian can be defined by a single term, retaining the index "\(K\)" to denote the component parallel to \(\eta_k\). Equations (8.51)-(8.58) define the algorithm matrix expressions \((F1)_e\), which are analytically differentiable by members of \((Q1)_e\).

The construction of certain terms involves differentiation with respect to the parameter \(u_k = \bar{m}_k\). Using the chain rule, and (8.46)-(8.47), as an exercise verify that
\[ \frac{\partial}{\partial \{MI\}_e} = \frac{\partial}{\partial \{MI\}} + \frac{\partial}{\partial \{MK\}} \frac{\partial \{MK\}}{\partial \{MI\}} + \frac{\partial}{\partial \{UK\}} \frac{\partial \{UK\}}{\partial \{MI\}} \] \hspace{1cm} (8.72)
Define the scalar \( \bar{D}_{KI} \) to be the element average value of \( \det J (\partial \eta_k / \partial x_i) \) appearing in (8.72). Further define \( K \) to signify the discrete free index, i.e., no summation implied, corresponding to \( \partial \eta_k / \partial x_i \) (8.70)-(8.71). The nonempty tensor matrix product jacobians for the finite-element algorithm statement (8.51)-(8.56), suppressing the subscript \( e \) throughout, for clarity, are

\[
\frac{\partial}{\partial (R) e} = \frac{\partial}{\partial (R)} - \left( \frac{m_k}{\bar{p}^2} \right) \frac{\partial}{\partial (UK)} \tag{8.73}
\]

\[
[JRR]_e = (DET)^T [A3000] + \nu_{J\ell} (ETAKJ)^T [A400K0] (DET)
+ \frac{\Delta t}{2} \nu_{RJ} \left[ (ETAKJ)^T [M40K0] \{UBARK\} \right.
- \left( \frac{m_k}{\bar{p}^2} \right) (ETAKJ)^T [M40K0] \{R\} \right]
\]

\[
[JRM]_e = \frac{\Delta t}{2} \left[ -(ETAKJ)^T [A30K0] + \nu_{J\ell} (ETAKJ)^T [A30KK] \right] \tag{8.74}
\]

\[
[JMI]_e = \frac{\Delta t}{2} \left[ -(ETAKJ)^T [A30K0] + \nu_{J\ell} (ETAKJ)^T [A40K00] \{ETAKJ\} \right]
\]

\[
[JMII]_e = (DET)^T [A3000] + \nu_{J\ell} (ETAKJ)^T [A40K00] (DET)
+ \frac{\Delta t}{2} \left[ -(ETAKJ)^T [A30K0] - \bar{D}_{KI} (MI)^T [A30K0] \frac{1}{\bar{p}} \right]
\]

\[
[JMIII]_e = \frac{\Delta t}{2} \bar{D}_{KI} \left( \frac{1}{\bar{p}} \right) \left[ -(MI)^T [A30K0] + \nu_{J\ell} (MI)^T [A40K00] \{ETAKJ\} \right]
\]

\[
[JMIP]_e = -\frac{\Delta t}{2} (ETAKJ)^T [A30K0] \tag{8.75}
\]

\[
[JMIII]_e = \frac{\Delta t}{2} \bar{D}_{KI} \left( \frac{1}{\bar{p}} \right) \left[ -(G + P)^T [A30K0] + \nu_{G\ell} (G)^T [A40K00] \{ETAKJ\} \right]
\]

\[
[JG]_e = (DET)^T [A3000] + \nu_{G\ell} (ETAKJ)^T [A40K00] (DET)
+ \frac{\Delta t}{2} \left[ -(UBARK)^T [A30K0] + \nu_{G\ell} (UBARK)^T [A40K00] \{ETAKJ\} \right]
\]

\[
[JGP]_e = -\frac{\Delta t}{2} (UBARK)^T [A30K0] \tag{8.76}
\]

\[
[JSI]_e = \frac{\Delta t}{2} (ETAKJ)^T [A40K00] \{UI\} \tag{8.77}
\]

\[
[JSQ]_e = \frac{\Delta t}{2} \bar{D}_{K\ell} (ETAKJ)^T [A30K0] \tag{8.78}
\]

\[
[JSR]_e = -\left( \frac{1}{\bar{p}} \right) \left[ (ETAKJ)^T [A40K00] \{UK\} \right]
\]

\[
[JPR]_e = -\left( \frac{1}{\bar{p}} \right) \left[ (ETAKJ)^T [A40K00] \{MK\} \right]
\]

\[
[JS]_e = -\left( \frac{1}{\bar{p}} \right) \left[ (ETAKJ)^T [A40K00] \{MK\} \right]
\]

\[
[JSJI]_e = (DET)^T [A3000] \tag{8.79}
\]

\[
[JSJII]_e = \frac{\Delta t}{2} \bar{D}_{KI} \left( \frac{1}{\bar{p}} \right) \left[ -(G + P)^T [A30K0] + \nu_{G\ell} (G)^T [A40K00] \{ETAKJ\} \right]
\]

\[
[JSJIII]_e = \frac{\Delta t}{2} \bar{D}_{KI} \left( \frac{1}{\bar{p}} \right) \left[ -(G + P)^T [A30K0] + \nu_{G\ell} (G)^T [A40K00] \{ETAKJ\} \right]
\]

\[
[JSJIV]_e = \frac{\Delta t}{2} \bar{D}_{KI} \left( \frac{1}{\bar{p}} \right) \left[ -(G + P)^T [A30K0] + \nu_{G\ell} (G)^T [A40K00] \{ETAKJ\} \right]
\]

The scalars \( \bar{m}_{i} \), \( \bar{p} \), and \( \bar{g} \) are element average values of \( \{MI\} \), \( \{R\} \), and \( \{G\} \), respectively on \( R_{e}^{1} \). In the formation of certain of these statements, the boolean index \( K \) in various A matrices has been permitted to facilitate differentiation by the last right contraction matrix. The elements of each Jacobian are computed on each finite-element domain \( R_{e}^{1} \), using the element matrices listed in Appendix B.1, and then assembled into the global form using the operator \( S_{e} \). In actual practice, the column matrix \( \{EQ\} \) is ordered on degrees of freedom at a node, for example \( \{\ldots, \delta R_{j}, \delta U_{j}, \delta E_{j}, \delta P_{j}, \delta R_{j+1}, \ldots\}^{T} \). Hence, the global tensor jacobian \( [\mathbf{J}] \) (8.59), is \( \alpha \)-block...
tridiagonal using the linear \((k = 1)\) finite-element basis, and \(a\)-block pentadiagonal for the quadratic \((k = 2)\) cardinal basis.

**Problems**
1. Verify (8.69) using (8.65)-(8.68).
2. Verify (8.72) and (8.73).
3. Derive (8.74).
4. Verify (8.75).
5. Verify (8.76).
6. Verify (8.77).

### 8.6 Comments on the Algorithm

Equations (8.51)-(8.58) are an exact statement of the elemental matrix equivalent of the finite-element algorithm statement, which upon assembly on \(\cup \mathbb{R}_2^2\) yields (8.33). While the master hypermatrix symbolology \([MP \cdots]\) is independent of the completeness of the semidefinite approximation subspace, i.e., the degree \(k\) of \((N_k(q))\) (8.30), the order of each square matrix is \((k + 1)^2\). In addition, since each matrix is a hypermatrix of degree \(p > 1\), there are \((k + 1)^p\) entries for each element of each matrix \([MP \cdots]\).

The associated inner product DO-loop to form \((F)\), especially for \(k > 1\), is excessive length on a scalar machine. The algorithm matrix statement can be simplified in this regard, with the commission of interpolation error only. Each boolean “zero” appearing in a hypermatrix \([MP \cdots]\) signifies the inner product with an element-dependent column (row) matrix \(\{\star\}\) corresponding to an interpolation \(\mathbb{R}_2^2\). For example, the lead term in each matrix statement \((F)\), (8.51)-(8.52), \((DET)_e^{T}[M3000]\_e\), in the instance of the linear transformation \(x = \eta\), the elements of \((DET)_e\) are each equal to a constant times the element measure. Hence, \((DET)_e^{T}(\{ONE\})^T\), where the elements of \(\{ONE\}\) are unity, which yields

\[
(DET)_e^{T}[M3000] = (DET)_e\_e\{ONE\}^T[M3000] = (DET)_e[M200]
\]

where \((DET)_e\) is the scalar measure (area, volume) of the finite-element domain. As presented in Sec. 8.9, for \(k = 1, n = 2\), and a general quadrilateral domain, the elements of \((DET)_e\) are each proportional to the element measure. Thus, the sum \((DET)_e \equiv (DET)^T\{ONE\}\) for a “decent” element aspect ratio can be approximated as

\[
(DET)_e^{T}[B3000] \approx (DET)_e[B200]
\]

with the commission of interpolation error only. This operation reduces by \(O(1)\) the number of computations required to form this term in \((F)\)

Thus, in this context, every hypermatrix inner product in (8.51)-(8.58) for reduction of hypermatrix degree through element averaging. This is appropriate only for data that are sufficiently smooth, which in general rules out

dependent variable, since nonsmooth solutions must be admitted. However, for decent discretizations \(\cup \mathbb{R}_2^2\), the geometric data should be eligible for averaging on sufficiently refined grids. Therefore, in (8.52) for example,

\[
(ETAKJ)_e^{T}[M40K00]\_e \rightarrow (ETAKJ)_e^{T}(DET)_e[M200]
\]

\[
(ETAKJ)_e^{T}[M40K00]U_BARL)_e \rightarrow (ETAKJ)_e^{T}(UBARL)_e[M30KL]
\]

Further, in (8.57), for example,

\[
(ETAKJ)_e^{T}(ETALI)_e^{T}[M50K00]\_e (KDIIFI)_e\rightarrow (ETAKJ)_e^{T}(ETAKL)_e^{T}(KDIIFI)_e[M30KL]
\]

Hence, no hypermatrix of degree \(p > 1\) is required formed or stored, provided the discretization of \(\mathbb{R}_2^2\) is of sufficient quality. The averaging of the grid data in the Jacobian tensor product matrix formulation, (8.74)-(8.79) is thus also appropriate.

A second point in the finite-element formulation (8.51)-(8.58) is that definition and use of the stress tensor \(q_j\), and heat flux vector \(q_j\) as dependent variables has yielded an algorithm statement devoid of second order derivatives in the generalized coordinates framework. Hence, the troublesome, sometimes destabilizing, mixed partial derivatives resulting from viscosity terms in the direct formulation are totally absent. The penalty for this dependent variable construction is a significant increase in the size of the \(a\)-block-banded tensor product Jacobian. Conversely, a general turbulent flow prediction can ostensibly be handled with ease, by expansion of the defining equations for \(q_j\) and \(q_j\). The fact that the algebraic “constitutive” equations for \(p\), \(q_j\), and \(q_j\) are handled directly within the weighted residuals algorithm framework lends an overall uniformity that simplifies construction.

Finally, a noniterative and direct steady-state form of the algorithm are each a special case of the presented formulation. The noniterative construction simply constitutes acceptance of the first solution \(\{Q(1)\}_{j+}\) of the Newton algorithm, using only solution of \((F)\) \(j+\) . Since the iteration index \(p > 0\) by definition \(\{Q(1)\}_{j+} = \{Q(1)\}_j\) therefore, \(\{Q(1)\}_{j+} = \{0\}\) throughout (8.51)-(8.58), and the corresponding expressions in brackets are not evaluated. Furthermore, \(\Delta t\rightarrow 0\) and the evaluation \(\star(\cdot)\) reduces to \(\cdot(\cdot)\). Since the terms in \((F)\) involving \(\nu_d\) have been eliminated, so are the corresponding terms in the tensor product Jacobians. The multiplier \(\Delta t/2\) remains appropriate in \(\{U\}_H\), and evaluations are made using \(\{Q(1)\}_{j+} = \{Q(1)\}_J\).

This noniterative procedure reduces the algorithm operations count by a significant factor, at the expense of removal of \(\nu_d\) from the construction and acceptance of the first Newton iterate, obtained using an approximate Jacobian. The direct steady-state algorithm is identical to the noniterative formulation except that \(\Delta t\rightarrow 0\) in Jacobians. This multiplier, common to all elements of \((F)\), can be divided out, using \(\Delta t\) as a scalar multiplier on \((DET)^T[A3000]\) in the self-coupling Jacobians \(\{Q\}_j\) for the initial-value dependent variables. This yields the tensor matrix generalization of the “approximate factorization” procedures devised for finite-difference methods (Beam and Warming, 1978; Briley and McDonald, 1977; Steger, 1981).
8.7 THEORETICAL ANALYSIS, ACCURACY
AND CONVERGENCE

As amply discussed and verified throughout this text, accuracy and convergence statements for finite-element algorithms are usually quantized as inequalities in Sobolev norms. The Navier-Stokes problem class for finite Reynolds number is nonlinear elliptic with initial-value character. For infinite Reynolds number (inviscid flow), the resulting (Euler) equation set is nonlinear hyperbolic. The available theoretical analyses are exact only for linear equations, and have already been presented; recall Secs. 2.8, 3.3, and 4.8. However, as presented in Chaps. 4 and 6, the convergence rates measured using controlled numerical experiments have provided indication that the linear theory is appropriate for nonlinear parabolic equations at least.

For a nonlinear, one-dimensional hyperbolic equation set (see Sec. 8.8), Baker (1982) measured convergence in $H^1$ and $E$ for a shocked (nonsmooth) flow using the developed finite-element algorithm. Figure 8.2 summarizes the data, which indicate that in the semidiscrete approximation norms $\|q_h\|_{H^1}$ and $\|q_h\|_E$, the solutions converge monotonically with discretization refinement for $10 < M < 400$. A modest slope distinction is evidenced with $\alpha$, the $k = 2$ data lies above the $k = 1$ results, but each appears consequentially independent of the degree of the approxima-

Figure 8.2 Semidiscrete approximation accuracy and convergence in $\|q_h\|_{H^1}$ and $\|q_h\|_E$, finite-element algorithm solution for Riemann shock tube. Solid symbols are Crank-Nicolson results, $M = 100$. From Baker (1982).

tion subspace $1 \leq k < 2$; see (8.30). This observation is in qualitative agreement with the theoretical convergence statement (4.136) for a linear hyperbolic equation. Furthermore, the $k = 1$ finite-element algorithm solution extremized each norm, in comparison to the Crank-Nicolson finite difference equivalent of the algorithm (recall Sec. 4.7), which considerably extends the problem class range over which this observation has been quantized. Viewing these data, the convergence character can be described in the form

$$\|q_h\|_{H^1}, E \leq C_1 \Delta^{p+1} + \cdots$$

which has been employed to define the order-of-accuracy ($p$) of a finite-difference scheme applied to nonsmooth solutions. As usual, $C_1$ is a constant independent of the (uniform) mesh measure $\Delta$, and $p = 2$ from the data of Fig. 8.2, for the $1 \leq k < 2$ bases.

The principal control in the order of accuracy of the finite-element algorithm statement (8.31) lies in determination of the dissipation parameter set $\beta_3$. Recall in Sec. 4.11, the Fourier stability analysis applied to the one-dimensional form of (8.1). The Fourier decomposition of the semidiscrete approximation is

$$q_h(i \Delta x, t) = Q_0 \exp \left( i \omega(t) \Delta x - i \Gamma_0 t \right)$$

where $\Gamma = \alpha + i \delta$, and $\alpha$ and $\delta$ are real numbers; $i = \sqrt{-1}$; $\omega = 2\pi/\lambda$ is the wave number for Fourier mode of wavelength $\lambda$; and $x = i \Delta x, j = 0, 1, 2, \ldots$ is represented by discrete intervals of (uniform) measure $\Delta x$. For the definition $\beta_3 = \alpha \Delta x^2$ in (8.31), where $\nu > 0$ is a scalar parameter, the Raymond and Gardner (1976) analysis yielded the $k = 1$ algorithm expansions for $\alpha$ and $\delta$ as [see (4.186)-(4.187)]

$$\alpha = 1 + \left( \frac{-1}{180} \frac{\nu^2}{12} \right) d^4 + O(d^6)$$

$$\delta = - \frac{\nu}{12} d^3 + O(d^5)$$

where $d = \omega \Delta x = 2\pi/n$, and $n$ is the discrete Fourier mode index, $\lambda = n \Delta x$, and $O$ indicates order. The semidiscrete solution $q_h$ can be made a sixth-order accurate approximation by eliminating the $O(d^4)$ term in (8.86), yielding $\nu = (15)^{-1/2}$. Correspondingly, $\delta < 0$ in (8.87) and an artificial dissipation mechanism becomes introduced.

The original analysis has been expanded (Baker, 1982) by redefining the dissipation parameter $\beta_3$ in the form

$$\beta_3 = \Delta x (\nu^4 \delta_1 + \nu^2 \delta_2)$$

where $\delta_1$ and $\delta_2$ are Kronecker delta-type functions, yielding $\nu^4$ operating on the time derivative, and $\nu^2$ operating on the spatial derivative term only, in the one-dimensional form of (8.1). Proceeding through the substitutions yields, for the $k = 1$ algorithm,

$$\sigma = 1 - d^2 (\nu^4 - \nu^2) + \left( - \frac{1}{180} \frac{\nu^4}{12} + (\nu^4 - \nu^2) (\nu^4) \right) + O(d^6)$$

(8.89)
\[ \delta = d(\nu^1 - \nu^2) - d^3 \left[ \frac{\nu^1}{12} - (\nu^1 - \nu^2)(\nu^1)^2 \right] + O(d^5) \] (8.90)

For the quadratic \((k = 2)\) algorithm, the form for (8.89) is

\[
\sigma = 1 - 4\nu^1 \nu^2 + \frac{d^2}{15} \left[ -14 + 184\nu^1 \nu^2 - 60(\nu^1)^2 + 240\nu^2(\nu^1)^3 \right]
+ \frac{2}{15} \left[ 542 \nu^1 \nu^2 + \frac{8}{3} (\nu^1)^2 - 622 \nu^2(\nu^1)^3 + 16(\nu^1)^4 - 64\nu^2(\nu^1)^5 \right] + O(d^5)
\] (8.91)

Setting \(\nu^1 \equiv \nu \equiv \nu^2\) in (8.89)-(8.90) yields the results of the original analysis. Enforcing sixth-order accuracy for (8.89)-(8.90) produces the constraint

\[ \nu^2 = \frac{d^2}{d^3} \left[ \frac{1}{180} - (\nu^1)^2 \right] + (\nu^1)^3 \] (8.92)

Figure 8.3 is a plot of (8.92) with \(n\) as a parameter. Sixth-order accuracy can be achieved only for \(\nu^1 > 0\); for any level, \(\nu^2\) ranges over an order of magnitude dependent upon \(n\), with the largest levels required for the shortest wavelengths. All data converge at the point \(\nu^1 = (15)^{-1/2} = \nu^2\). Defining \(\nu^1 = 0\) in (8.89) or (8.91) renders both the \(k = 1\) and \(k = 2\) algorithm constructions, for the semidiscrete approximation \(q^2\), a second-order accurate representation of the analytical solution for \(\nu^2 \neq 0\).

The expansion of (8.88) for the multidimensional, multidependent variable, generalized coordinates definition for the dissipation parameter \(\beta_k^2\) is

\[ \beta_k^2 = (\det J)(\nu_k^1 \delta_k^1 + \nu_k^2 \delta_k^2) \] (8.93)

In (8.93), \(\det J\) is the measure of \(R_\alpha^2\), subscript \(\alpha\) denotes the appropriate member of \(q_\alpha^2\), and the parameter vectors \(\nu_k^\alpha\) are expressed in terms of scalar components \(\nu_k^\beta\) in the \(x_\beta\) coordinate system.

Problems
1. Verify (8.86)-(8.87) (see Sec. 4.11).
2. Verify (8.89).

8.8 ALGORITHM CONSTRUCTION FOR QUASI-ONE-DIMENSIONAL FLOW

Most formulational aspects of the generalized coordinates, Navier-Stokes finite-element algorithm construction can best be exposed by examination of a quasi-one-dimensional inviscid flow. Defining the convection velocity \(u = m/\rho\), and the flow cross-sectional area as \(A(x)\), we form from (8.1)-(8.4) the governing differential equation set

\[ L(p) = \frac{\partial p}{\partial t} + \frac{\partial}{\partial x} [m] + \rho \frac{d}{dx} \ln A = 0 \] (8.94)

\[ L(m) = \frac{\partial m}{\partial t} + \frac{\partial}{\partial x} [um + p] + mu \frac{d}{dx} \ln A = 0 \] (8.95)

\[ L(q) = \frac{\partial q}{\partial t} + \frac{\partial}{\partial x} [ug + up] + u(g + p) \frac{d}{dx} \ln A = 0 \] (8.96)

\[ L(p) = p - (\gamma - 1)[g - \frac{1}{2} um] = 0 \] (8.97)

The algorithm statement requires the metric data \((DET)_e\) and \((ETAJ)_e\). For the affine coordinate transformation \(x_\beta = \eta_\beta\), and with the \(\eta_j\) origin at the element centroid, the members of the cardinal bases \((N^k_j(\eta))\), \(1 \leq k \leq 2\), are

\[ (N^k_1(\eta)) = \frac{1}{2} \left\{ \begin{array}{c} 1 - \eta \ \eta \\ 1 + \eta \end{array} \right\} \] (8.98)

\[ (N^k_2(\eta)) = \frac{1}{2} \left\{ \begin{array}{c} -\eta(1 - \eta) \\ 2(1 - \eta)(1 + \eta) \\ \eta(1 + \eta) \end{array} \right\} \] (8.99)
By definition, \( x_i = (N_k(x)) (X)_{\varepsilon} \), and \( J = \det \{ \partial x_i / \partial \eta_j \} \). Denoting the elements of \( (X)_{\varepsilon} \) as the left \( (L) \), right \( (R) \) and middle \( (M) \), coordinates of \( R^2 \), where by definition \( M = \frac{1}{2} (R + L) \), it is easy to verify that for
\[
k = 1: \quad DET_x = \frac{1}{2} (R - L)
k = 2: \quad DET_x = \frac{1}{2} (R - L) - \eta (R + L - 2M)
\] (8.100)

Since the element measure definition is \( \Delta \varepsilon = R - L \), then \( DET_x = \Delta \varepsilon / 2 \) for each basis. By the same procedure, the sole nonvanishing element of \( \{ ETAKJ \} \) is in the \((1, 1)\) location with a value of unity.

Therefore, the finite-element algorithm statement becomes quite simplified for the quasi-one-dimensional situation. It is a suggested exercise to show that the algorithm statement (8.51)–(8.54) is
\[
\begin{align*}
\{ FR \} & = \Delta \varepsilon \{ [A200] + \nu^2 [A210] \} [R]^{j+1} + \frac{\Delta t}{2} \left( -[A210] \{ M \} + \nu^2 \left( \partial^T [A3011] \{ R \} \right) \right) + \{ A \}^T \{ [A41000] \} \{ \Omega \} \{ R \}^{j+1} = \{ 0 \} \quad (8.101) \\
\{ FM \} & = \Delta \varepsilon \{ [A200] + \nu^2 [A210] \} [M]^{j+1} + \frac{\Delta t}{2} \left( -[R] \{ [A3010] \} \{ M \} - [A210] \{ P \} + \nu^2 \left( \partial \right) [A3011] \{ M \} + \{ A \}^T \{ [A41000] \} \{ \Omega \} \{ M \} \right) \{ R \}^{j+1} = \{ 0 \} \quad (8.102) \\
\{ FG \} & = \Delta \varepsilon \{ [A200] + \nu^2 [A210] \} \{ G \}^{j+1} + \frac{\Delta t}{2} \left( -[R] \{ [A3010] \} \{ G \} + \{ P \} \right) + \nu^2 \left( \partial \right) [A3011] \{ G \} + \{ A \}^T \{ [A41000] \} \{ \Omega \} \{ G \} \{ R \}^{j+1} = \{ 0 \} \quad (8.103) \\
\{ FP \} & = \Delta \varepsilon \{ [A200] \} \{ P \} - \Delta \varepsilon (\gamma - 1) \{ [A40000] \} \{ G \} - \frac{\Delta t}{2} \left( [R] \{ [A3010] \} \{ M \} \right) \{ R \}^{j+1} = \{ 0 \} \quad (8.104)
\end{align*}

It is an elementary task to differentiate (8.101)–(8.104), to form the Newton algorithm jacobian contributions, (8.74)–(8.77), and to show that
\[
\begin{align*}
\frac{\partial \{ FR \}}{\partial \{ R \}} = \Delta \varepsilon \{ [A200] + \nu^2 [A210] \} + \frac{\Delta t}{2} \left( 2 \nu^2 \left( \partial \right) [A3011] \right) - \left( \frac{\vec{m}}{\vec{p}^2} \right) \{ [A3110] \} + \{ A \}^T \{ [A41000] \} \left( \frac{\vec{m}}{\vec{p}^2} \right) \{ [R] \}^{j+1} \quad (8.105)
\end{align*}
\]
\[
\begin{align*}
\frac{\partial \{ JMR \}}{\partial \{ R \}} = \Delta \varepsilon \left( \left\{ [A3010] + \nu^2 [A3110] \right\} \{ R \}^{j+1} \right) \quad (8.106)
\end{align*}
\]
\[
\begin{align*}
\frac{\partial \{ JMR \}}{\partial \{ R \}} = \Delta \varepsilon \left( \left\{ [A3010] + \nu^2 [A3110] \right\} \{ R \}^{j+1} \right) \quad (8.107)
\end{align*}
\]
\[
\begin{align*}
\frac{\partial \{ JMR \}}{\partial \{ R \}} = \Delta \varepsilon \left( \left\{ [A3010] + \nu^2 [A3110] \right\} \{ R \}^{j+1} \right) \quad (8.108)
\end{align*}
\]

In (8.106)–(8.108), the superscript bar on \( m \) and \( \rho \) indicates the element average value, and absolute value when multiplied by \( \nu^2 \). The elements of \( \{ A \} \) are the nodal values of \( \ln A(x) \). The defined standard matrices and hypematrices are listed in Appendix B.1 for both cardinal basis formulations, \( 1 \leq k \leq 2 \).

It is important to detail the construction of the jacobian of the Newton iteration algorithm (8.34), especially since the multidimensional algorithm employs basically
one-dimensional constructions in forming the matrix tensor product Jacobian (8.59). Symbolically, using the assembly operator \( S_e \), the Newton statement (8.34) is the form

\[
S_e \left[ \begin{array}{ccc} [JRR]_e & [JRM]_e & 5R \\ [UMR]_e & [JMM]_e & \delta M \\ [JGR]_e & [JGM]_e & \delta G \\ [JPR]_e & [JPM]_e & \delta P \end{array} \right] = - \left[ \begin{array}{c} FR \\ FM \\ FG \\ FP \end{array} \right] \quad (8.109)
\]

with the various element Jacobians given by (8.105)-(8.108). In actual practice, to minimize bandwidth, the global vector \( \delta Q \) is ordered by degrees of freedom at node points, e.g.,

\[
(\delta Q)^T = \{ \ldots, R_{j-1}, M_{j-1}, G_{j-1}, P_{j-1}, R_j, M_j, G_j, P_j, R_{j+1}, M_{j+1}, G_{j+1}, P_{j+1}, \ldots \} \quad (8.110)
\]

For exposition, the lead matrix in each elemental Jacobian of self-coupling \([JQ]_e \) is \( A_e[A200] \). Assembling over the element pair sharing node coordinate \( x_j \), for the \( \{N_1\} \) algorithm, yields the tridiagonal recursion relationship

\[
S_e[A_e[A200]] = S_e \left[ \begin{array}{ccc} 2 & 1 \\ 1 & 2 \\ \Delta j-1 & \frac{1}{6} & \Delta j+1 \end{array} \right] \quad (8.111)
\]

assuming \( \Delta j \equiv (x_j - x_{j-1}) \) and \( \Delta j+1 \equiv (x_j+1 - x_j) \) are distinct. Hence, define the elements of \( S_e[JQ] \) as

\[
S_e[A_e[A200]] = \{ \ldots, 200j-1, 200j, 200j+1, \ldots \} \quad (8.112)
\]

and the numerical values are given in (8.111). In a similar manner, noting the first term in \([JMR]_e \) involves \( M_j[A3010] \), define the elements of the assembled form as

\[
S_e \left[ -\frac{\Delta t}{2} \left( \frac{\vec{m}}{\rho^2} \right)_e [A3010] \right] = \{ \ldots, M3010j-1, M3010j, M3010j+1, \ldots \} \quad (8.113)
\]

Both forms (8.112) and (8.113) would extend to \( (+)_{j-2} \) and \( (+)_{j+2} \) should the \( \{N_2\} \) basis be employed rather than \( \{N_1\} \).

Using the definitions (8.112)-(8.113), the Newton algorithm statement (8.109) assembled on degrees of freedom at node coordinates, is the \( \alpha \)-block partitioned matrix equation:

\[
\begin{bmatrix}
R_{j-1} \\
M_{j-1} \\
G_{j-1} \\
P_{j-1} \\
M_j \\
G_j \\
P_j \\
M_{j+1} \\
G_{j+1} \\
P_{j+1}
\end{bmatrix} = - (\delta Q)
\]

(8.114)

In (8.114), each symbol corresponds to only the first term in any specific element Jacobian (8.105)-(8.108). The prefixes \( M, G, U \), on the 3010 and 3000 terms, indicate the occurrence of explicit nonlinearity in \( J^* \). Note that the block structure is identical in appearance in each partition \( \{1\}_{j-1}, \{1\}_j \), and \( \{1\}_{j+1} \). However, the entries are distinct, e.g.,

\[
200j-1 = \Delta j-1 \quad 200j = 2(\Delta j+1 + \Delta j-1) \quad 200j+1 = \Delta j+1
\]

For the quadratic basis algorithm construction, the symbols in (8.114) extend to \( \pm j \). The scalar bandwidth for the \( \{N_1\} \) algorithm statement (8.114) is 12, since both the energy and pressure are fully coupled throughout the dependent variable set.

**Problems**

1. Derive (8.94)-(8.97) from (8.1)-(8.4).
2. Verify the one-dimensional basis \( \{N_2(e)\} \) given in (8.98)-(8.99).
3. Verify (8.100).
6. Derive the elements of (8.114) using (8.111)-(8.113).

**8.9 ACCURACY AND CONVERGENCE, MIXED ONE-DIMENSIONAL FLOWS**

The principal requirement of any quasi, one-dimensional solution is to quantize accuracy and convergence performance of the finite-element algorithm for mixed
subsonic-supersonic flows. Shock sharpness and associated undershoot/overshoot are readily assimilated measures of solution acceptability. The Riemann shock tube simulation (Shapiro, 1953, p. 1007) is a well-suited problem definition, since the resultant flow structure is richly endowed with discontinuities and sharp field gradients interspersed with planar plateau regions. The interesting case, with unique stagnation sound speeds in the two chambers initially separated by the diaphragm, has been exhaustively examined for algorithm performance in the finite-difference literature (van Leer, 1979; Zalesak, 1980). In particular, Sod (1978) compares the results produced by a dozen finite-difference algorithms, based selectively on both Lagrangian and mixed Lagrangian-Eulerian frameworks, for a single Riemann shock tube specification.

Figure 8.4 $M = 400$, $k = 1$ finite-element algorithm solution, Riemann shock tube, $t = 0.14154$ s, $\nu_\alpha = \nu = \nu_\beta$, $\nu_\gamma = 0$. (---) Denotes initial conditions.

For the finite-element algorithm assessment, Baker (1982), the diaphragm is placed midway in a duct of uniform cross section, with the unit length discretized into $12 \leq M \leq 400$ finite elements $R_e$ of uniform measure $\Delta$. The $k = 1$, $M = 100$ discretization corresponds to the definition of Sod (1978), with initial condition specifications $u(x) = 0$, $p = 1 = \rho$ on $0 \leq x \leq 0.5$, $p = 0.1$ and $\rho = 0.125$ on $0.5 \leq x \leq 1.0$, and $u = 1.4$. Figure 8.4 graphs the $k = 1$ solution variable set $\{Q(t \Delta t)\}$ at $t = 0.14154$ s, as obtained using the Raymond-Gardner order-of-accuracy optimized parameter set $\nu_\delta = \nu = \nu_\beta = 0$, $\nu = 0$, and $\rho = (15)^{-1/2}$. Each symbol corresponds to a nodal coordinate of $q_{\text{eq}}$, and the dashed lines denote the initial conditions. The shock is centered at $x = 0.75$, the contact discontinuity is centered at $x = 0.62$, and the rarefaction wave lies upstream of $x = 0.5$, the diaphragm location. For comparison, Fig. 8.5 is a graph of the $M = 99$ solution for $p^b_\text{eq}$ and $p^b$, on the region $0.25 \leq x \leq 0.75$, as generated by the Lagrangian-rezone Eulerian MUSCL finite-difference algorithm of van Leer (1979). The analytical solution is shown as solid lines. The $k = 1$ finite-element solution is considerably smoother, with less well-defined gradients and plateaus, and with excessive undershoot/overshoot about the shock. Figure 8.6 is an improved $k = 1$, $M = 200$, $(\nu_\beta)$ algorithm solution, obtained with the "numerically optimized" dissipation parameter set $\nu_\delta = \nu_\beta = 0$, $\nu_\beta = \nu = 0$, and $\rho = (15)^{-1/2}$ and $1 \leq \alpha \leq 3$. Each of the characteristic Riemann solution features is accurately predicted including planar plateau regions and a crisply defined shock with negligible overshoot.

The solution fields of principal engineering interest for the Riemann problem are velocity and internal energy (temperature) distributions. Figure 8.7a graphs the $M = 99$ MUSCL code solution and the analytical solution (solid line). The $\nu_\beta$ optimized, $k = 1$, $M = 200$ and $M = 100$ finite-element solutions are graphed in Figs. 8.7b-8.7c, respectively. In addition, Fig. 8.7d shows the $M = 100$ solution obtained using the
high temperature plateau are nominally identical. The MUSCL code interpolates the contact discontinuity over five domains, while the $k = 1$ data has smeared it over nine finite elements. Considering that the eulerian finite element algorithm is not at all "hard-wired" for this problem, this accuracy is quite acceptable.

The accuracy of the $k = 2$ finite-element algorithm for the Riemann problem is nominally identical. The optimized dissipation parameter set was determined as $
u_0^+ = 0$, and $
u_0^+ = \nu (1/4, 3/4, 1/2)$ (Baker, 1982). Figure 8.8 graphs the resulting

Crank-Nicolson finite-difference algorithm equivalent of the $k = 1$ finite-element algorithm with $\nu_0^+ = 0$ and $\nu_0^+ = \nu (1, 1, 1)$. Increasing the dissipation level $\nu_0^+$ would moderate the illustrated excessive shock overshoot, at the expense of further diffusing the shock over more than the present five domains. In comparison, the $M = 100$, $k = 1$ finite-element algorithm data interpolate the shock over only two element domains with negligible overshoot, for the identical CPU and main memory requirements. The $M = 100$, $k = 1$ finite-element data (Fig. 8.7c), compare favorably with the MUSCL code data. The velocity resolution of the shock and the planarity in the

---

Figure 8.6 $M = 200$, $k = 1$ finite-element solution, Riemann shock tube $\nu_0^+ = \nu (3/8, 0, 1/4)$, $\nu_0^+ = \nu (3/4, 2, 1)$, $t = 0.14154 \pm (\cdots \cdots)$ Denotes initial conditions.

Figure 8.7 Finite-element and finite-difference algorithm solution comparisons, Riemann shock tube; $t = 0.14154 \pm (\cdots \cdots)$ MUSCL code solution of van Leer (1978); (---) analytical solution, reprinted with permission; $M = 200$, $k = 1$ finite-element solution, $\nu_{opt}$. 
\[ \nu_x = 0 \text{ is "optimal," the algorithm results are not measurable degraded (see Fig. 8.96).} \]

The Newton iteration algorithm Jacobian construction (8.114) approximates the theoretical quadratic convergence rate. For example, Table 8.1 summarizes location and magnitude of extremum elements of \( \|\mathbf{Q}\| \) for a typical integration step.

The sharply defined solutions, as obtained using minimal levels for \( \nu_x \), typically require 80 steps and 250 iterations to reach \( t = 0.14154 \text{ s} \), at a Courant number \( (C = \mu + \sigma \Delta t/\Delta x) \) of approximately 0.33. Since the algorithm is implicit, the sole constraint on Courant number is accuracy. For these results, a Richardson extrapolation step at \( t = 0.14154 \text{ s} \) confirmed that the significant digit in \( \|\mathbf{Q}\| \) was unaffected.

solution \( \{Q/(n \Delta t)\} \) at \( t = 0.14154 \text{ s} \), using an \( M = 50 \) discretization. Figure 8.9s graphs the companion solutions for velocity and temperature, where the solid lines are traces of the exact solution. In comparison to Fig. 8.7a-c, accuracy is excellent with a noticeable improvement in definition of the contact discontinuity. For the case of the diagonalized finite-difference equivalent of the \( k = 2 \) algorithm, since

\[ \nu_x = 0, \quad \nu_y = \nu \{1/4, 3/4, 1/2\}. \]
by temporal truncation error. This of course is a requirement to accurately estimate the semidiscrete (finite-element) approximation error. Figure 8.2 graphs the algorithm semidiscrete convergence characteristics, measured in both $H^1$ and $E$, as discussed in Sec. 8.7.

Solution efficiency is principally determined by discretization ($M$) and integration time step ($C$), in concert with the convergence requirement for the Newton iteration algorithm. Table 8.2 summarizes for the $M = 100, k = 1$ Riemann solution, the effect of decreasing solution cost by increasing $C$. Solution cost is nominally halved by doubling the Courant number, to a certain point. Thereafter, diminishing returns are encountered as additional Newton iterations are required for the same convergence level. In all data for $C > 0.33$, the significant digit in the semidiscrete $E$ norm is directly affected by temporal truncation error. Hence, solution accuracy is degraded monotonically with increasing $C$. It is noteworthy that the $k = 1$ finite-element method accuracy in $E$, obtained at $C = 1.41$, is uniformly superior to the $C = 0.33$ finite-difference solution (see Fig. 8.2), and was obtained at 40 percent of the computer cost.

Since the finite-element method is fully implicit, the entirety of $R^1$ is in continuous communication, even those regions where the flow is locally supersonic. An interesting test occurs for a mixed-flow situation, wherein boundary data modifications in a subsonic region require an adjustment to flow in a supersonic region. For example, off-design operation of a de Laval nozzle yields a strong shock in the diverging section, such that the specified mass flow can diffuse to the exit chamber pressure. A modest increase in the exit pressure requires the shock to relocate upstream of its original location, wherein the flow is uniformly supersonic. During this period of adjustment, the supersonic flow upstream of the shock must not respond to

**Table 8.1 Newton iteration convergence in (δQI) Riemann shock tube ($M = 100, k = 1, ε = 0.001$)**

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$[δR]_{max}$</th>
<th>Node</th>
<th>$[δM]_{max}$</th>
<th>Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.026</td>
<td>70</td>
<td>0.051</td>
<td>70</td>
</tr>
<tr>
<td>2</td>
<td>0.0018</td>
<td>69</td>
<td>-0.0028</td>
<td>69</td>
</tr>
<tr>
<td>3</td>
<td>0.000082</td>
<td>69</td>
<td>-0.0038</td>
<td>69</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$[δG]_{max}$</th>
<th>Node</th>
<th>$[δP]_{max}$</th>
<th>Node</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.14</td>
<td>70</td>
<td>0.049</td>
<td>70</td>
</tr>
<tr>
<td>2</td>
<td>-0.0072</td>
<td>70</td>
<td>0.0048</td>
<td>71</td>
</tr>
<tr>
<td>3</td>
<td>0.00032</td>
<td>70</td>
<td>0.0026</td>
<td>68</td>
</tr>
</tbody>
</table>

**Table 8.2 Accuracy and efficiency summary, Riemann shock tube ($M = 100, k = 1, ε = 0.001$)**

<table>
<thead>
<tr>
<th>Courant number $C$</th>
<th>Number of integration time steps</th>
<th>Number of algorithm passes</th>
<th>GO step CPU</th>
<th>Energy norm $\delta E_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.33</td>
<td>80</td>
<td>229</td>
<td>1.00</td>
<td>1.54 2.43 17.9</td>
</tr>
<tr>
<td>0.44</td>
<td>60</td>
<td>180</td>
<td>0.79</td>
<td>1.50 2.39 17.4</td>
</tr>
<tr>
<td>0.57</td>
<td>40</td>
<td>121</td>
<td>0.54</td>
<td>1.46 2.29 16.8</td>
</tr>
<tr>
<td>0.90</td>
<td>30</td>
<td>92</td>
<td>0.42</td>
<td>1.41 2.15 16.0</td>
</tr>
<tr>
<td>1.41</td>
<td>20</td>
<td>82</td>
<td>0.38</td>
<td>1.34 1.90 14.4</td>
</tr>
</tbody>
</table>
the subsonic exit pressure level, the shock must progressively weaken as it moves upstream into regions of smaller cross section, and the flow downstream of the shock must diffuse smoothly to the new exit condition.

Figure 8.10 summarizes an $M = 74$, $k = 1$ algorithm unsteady solution for off-design nozzle flow with $M_1 = 1.35$ before the shock, using $v_{\text{opt}}$ and $\frac{1}{2}v_{\text{opt}}$. The solid line is the exact solution, and the $\frac{1}{2}v_{\text{opt}}$ solution is in excellent agreement on shock strength and definition. It is interesting how the momentum solution inter-

![Momentum](image1)

![Mach number](image2)

Figure 8.10 Finite-element algorithm steady-state solution, off-design de Laval nozzle flow; $M = 74$, $k = 1$; (a) $T = v_{\text{opt}}$; (b) $T = \frac{1}{2}v_{\text{opt}}$.

![Density](image3)

![Momentum](image4)

![Pressure](image5)

![Mach number](image6)

Figure 8.11 $M = 74$, $k = 1$ algorithm solution for mixed flow in a variable cross-sectional duct, $T_{\text{in}} = \frac{1}{2}v_{\text{opt}}$. (——) initial condition.

polates the shock discontinuity, which of course must be of finite width. The subsonic inlet boundary conditions are specified $\rho$, $m$, and $g$, with $p$ computed from (5.4). The subsonic outlet condition is $p$ specified, and vanishing normal derivatives for $\rho$, $m$, and $g$. Using this solution as an initial condition, the exit pressure was raised 15 percent and held fixed. Figure 8.11 summarizes the $k = 1$ algorithm solution at the new steady-state, where the solid lines show the initial conditions. The flow has adjusted to the modified subsonic exit pressure, with a new shock Mach number of $M_1 = 1.15$. The flow upstream of the shock is unaltered from the initial conditions, and a smooth subsonic expansion to exit conditions is predicted downstream. The
8.10 TWO-DIMENSIONAL FORMULATION, LINEAR TENSOR PRODUCT BASIS

As noted in Sec. 8.1, a wide variety of methodologies exist for approximate generation of the inverse coordinate transformation \( \eta_i = \eta_i(x_j) \); see (8.37). For example, Fig. 8.13 illustrates such body-fitted coordinate systems, for various two-dimensional aerodynamic configurations, as generated using Poisson equation techniques (Thames et al., 1977). The principal requirement is to render contours coincident with aerodynamic surfaces, symmetry, or periodicity boundaries, to be coordinate curves of the \( \eta_i \) system. The next requirement, from the standpoint of solution accuracy, is to avoid distorted computational cells in critical flow regions, e.g., about stagnation points.

With respect to the developed finite-element algorithm, each of the illustrated

\[ \text{Figure 8.12 \( M = 37, k = 2 \) algorithm solution for mixed flow in a variable cross-sectional duct,} \]
\[ \text{\( v_k = \frac{1}{2} v_{\text{opt}} \): (---) initial condition.} \]

shock is interpolated across four elements with no overshoot. Figure 8.12 summarizes the comparison \( M = 37, k = 2 \) algorithm solution obtained using \( v_k = \frac{1}{2} v_{\text{opt}} \), and the solution accuracies are indistinguishable.

\[ \text{Figure 8.13 Examples of computational grid transformations for two-dimensional aerodynamic flow predictions. (a) Turbine cascade, From Ghia and Ghia (NASA, 1980, p. 302). (b) Highly cambered airfoil and trailing edge close-up, From Sorensen and Steger (NASA, 1980, p. 456).} \]
(8.116)

\[
\{ETAKJ\}_e = \frac{1}{2}
\begin{cases}
(Y_4 - Y_1, Y_3 - Y_2, Y_3 - Y_2, Y_4 - Y_1)^T_e, & (1, 1) \\
(X_1 - X_4, X_2 - X_3, X_2 - X_3, X_1 - X_4)^T_e, & (1, 2) \\
(Y_1 - Y_2, Y_1 - Y_2, Y_4 - Y_3, Y_4 - Y_3)^T_e, & (2, 1) \\
(X_2 - X_1, X_2 - X_1, X_3 - X_4, X_3 - X_4)^T_e, & (2, 2)
\end{cases}
\]

The numbers in parenthesis indicate the indices \((K, J)\) in \(\{ETAKJ\}_e\). The corresponding definition for \(\det [J]_e\) is

\[
\det [J]_e = \{N_1\}_e^T \{DET\}_e
\]

yielding

(8.117)

\[
\{DET\}_e = \frac{1}{4}
\begin{cases}
(X_2 - X_1)(Y_4 - Y_1) - (X_4 - X_1)(Y_2 - Y_1) \\
(X_2 - X_1)(Y_3 - Y_2) - (X_3 - X_2)(Y_2 - Y_1) \\
(X_3 - X_4)(Y_3 - Y_2) - (X_3 - X_2)(Y_3 - Y_4) \\
(X_3 - X_4)(Y_4 - Y_1) - (X_4 - X_1)(Y_3 - Y_4)
\end{cases}
\]

The contravariant convection velocity approximation \(\vec{u}^h_k\), (8.46), on the domain \(R^2\) is defined as

(8.118)

\[
\vec{u}^h_k \equiv \{N_1\}_e^T \{UBARK\}_e
\]

and the algebra yields

(8.119)

\[
\{UBARK\}_e = \{ETAKJ\}_e \{N_1\}_e^T \{U\}_e
\]

Since (8.120) is defined on a nodal basis, the elements of \(\{N_1\}_e\) reduce to the kronecker delta, hence,

(8.121)

\[
\{UBARK\}_e = \{U\}_e \{ETAKJ\}_e
\]

Summation is implied over \(J\), and at the nodes \(\{U\}_e \equiv MJR\), that is, \(u_j = m_j J\).

The tensor product jacobian formulation, given for the quasi-one-dimensional flow, is modified only to account for alignment with either the \(n_1\) or \(n_2\) coordinate axis. The cardinal basis remains as given in (8.98), and \(DET\) is equal to one-half the element measure (8.100). Hence, for (8.74)-(8.79), and using Pythagoras' rule,

(8.122)

\[
\{DET\}_e = \frac{1}{2} \sqrt{(XKR - XKL)^2 + (YKR - YKL)^2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\]

where \(R\) and \(L\) denote right and left, respectively, for the specific \(n\) alignment. Correspondingly, the elements of \(\{ETAKJ\}_e\) are equal to the direction cosines of the local affine coordinate transformation between \(x_1\) and \(n_1\). Denoting \(\theta^k\) as the angle between the global coordinate \(x_1\), and the local \(n_k\) axis on \(R^2\) (see Fig. 8.14), then

(8.123)

\[
\{ETAKJ\}_e = \begin{bmatrix}
\cos \theta^k, \sin \theta^k \\
-\sin \theta^k, \cos \theta^k
\end{bmatrix}^T_e
\]

\[
(1, 1) \\
(1, 2)
\]

\[
\begin{bmatrix}
\{ETAKJ\}_e = \begin{bmatrix}
1 \\
1
\end{bmatrix}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\{ETAKJ\}_e = \begin{bmatrix}
\cos \theta^k, \sin \theta^k \\
-\sin \theta^k, \cos \theta^k
\end{bmatrix}^T
\end{bmatrix}
\]

\[
(1, 1) \\
(1, 2)
\]

\[
\begin{bmatrix}
\{ETAKJ\}_e = \begin{bmatrix}
1 \\
1
\end{bmatrix}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\{ETAKJ\}_e = \begin{bmatrix}
\cos \theta^k, \sin \theta^k \\
-\sin \theta^k, \cos \theta^k
\end{bmatrix}^T
\end{bmatrix}
\]

\[
(1, 1) \\
(1, 2)
\]

\[
\begin{bmatrix}
\{ETAKJ\}_e = \begin{bmatrix}
1 \\
1
\end{bmatrix}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\{ETAKJ\}_e = \begin{bmatrix}
\cos \theta^k, \sin \theta^k \\
-\sin \theta^k, \cos \theta^k
\end{bmatrix}^T
\end{bmatrix}
\]

\[
(1, 1) \\
(1, 2)
\]
The numbers in parentheses again indicate the indices \((K, J)\). Using (8.123), the element node definition for convection velocity in each \([\eta_1]\) is [cf. (8.121)]

\[
\{UBARK\}_e = U J_e \{ETAKJ\}_e
\]  

(8.124)

with \(\{ETAKJ\}_e\) provided by (8.123).

No further comments are required concerning evaluation of the algorithm matrices \(\{FI\}\), (8.50)-(8.58). The Newton Jacobian matrix tensor product solution is [cf. (8.59)-(8.62)]

\[
\{J_{1}\} \{FI\}_e = \{S\} \{Q\}_e
\]  

(8.125)

The generalization of (8.109) for the two-step solution procedure (8.125) is best illustrated by assuming the \(x_i\) and \(\eta_n\) coordinate systems are parallel, hence, \(\{ETAKJ\}_e = (0, 1)\). For the first solution sweep in (8.125), assumed to be executed parallel to the \(\eta_1\) coordinate direction, the Newton statement (8.109) takes the form

\[
\begin{bmatrix}
\{JRR\}_e \\
\{JMR1\}_e \\
\{JMR2\}_e \\
\{JGR\}_e \\
\{JPR\}_e \\
\{JSUR\}_e \\
\{JVR\}_e \\
\{JVR1\}_e
\end{bmatrix}
\begin{bmatrix}
\{JRM2\}_e \\
\{JM1M2\}_e \\
\{JM2M2\}_e \\
\{JGM1\}_e \\
\{JPM1\}_e \\
\{JSJM1\}_e \\
\{JQI\}_e \\
\{JQI1\}_e
\end{bmatrix}
\begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
= \{FI\}_e
\]  

(8.126)

For the second sweep parallel to \(\eta_2\), (8.109) is the form

\[
\begin{bmatrix}
\{JRR\}_e \\
\{JMR1\}_e \\
\{JMR2\}_e \\
\{JGR\}_e \\
\{JPR\}_e \\
\{JSUR\}_e \\
\{JVR\}_e \\
\{JVR1\}_e
\end{bmatrix}
\begin{bmatrix}
\{JRM2\}_e \\
\{JM1M2\}_e \\
\{JM2M2\}_e \\
\{JGM1\}_e \\
\{JPM1\}_e \\
\{JSJM1\}_e \\
\{JQI\}_e \\
\{JQI1\}_e
\end{bmatrix}
\begin{bmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{bmatrix}
= \{FI\}_e
\]  

(8.127)

In (8.126)-(8.127), the tensor indices in \(Q\) and \(QI\) range \(1 \leq (I, J) \leq 2\), and the zeros indicate no contribution on that sweep.

The individual contributions to \(S_e \{Q\}_e\), and the other terms in (8.126)-(8.127), are assembled as illustrated by (8.114). The maximum band width of the Jacobian \(\alpha\)-block structure is nominally 30, which is three times the number of nodal dependent variables, i.e., degrees of freedom per node. Rearrangement of the order of variables in \(\{QI\}\) can reduce this number for each specific sweep.

**Problems**

1. Derive (8.116) from (8.115) and (8.39).
2. Establish (8.118) from (8.117) and (8.39).
3. Verify (8.120)-(8.121).
4. Expand (8.122) and verify its appropriateness for the tensor matrix product Jacobians (8.74)-(8.79).
5. Verify (8.126) and expand it in the form (8.114).
6. Verify (8.127) and expand it in the form (8.114).

**8.11 TWO-DIMENSIONAL INVIScid AND VISCOUS FLOW**

Several issues basic to accuracy and convergence aspects of the tensor matrix product formulation of the generalized coordinates finite-element Navier-Stokes algorithm are reported by Baker (1982). Particular aspects studied include application of gradient boundary conditions, Newton algorithm convergence, prediction accuracy for shocks oblique to the mesh and computed in nonprincipal coordinates, and a shock-boundary layer interaction at \(Re = 10^5\).
The test problem definition is the two-dimensional generalization of the Riemann shock tube discussed in Sec. 8.9. By using the linear tensor product basis ($N^T$), solutions were generated on discretizations of $M = 32 \times N$, where $4 \leq N \leq 20$ dependent upon whether the problem was defined as inviscid or viscous. For reference, Fig. 8.15 graphs the coarse-grid ($M = 32$), one-dimensional $k = 1$ algorithm solution prediction of velocity and internal energy for the Riemann problem. The solid lines are traces of the $M = 200$, $k = 1$ solution from Fig. 8.7b, and each open symbol is a nodal solution value obtained using $v_0^2 = v_{opt}$. Interestingly, on this coarse grid, the importance of $v_0^2 > 0$ on accuracy becomes diminished. The solid symbols in Fig. 8.15 are the comparison nodal values, as obtained using $v_0^2 = 0$, $v_0^2 = \nu (3/4, 2, 1)$ in the $k = 1$ algorithm solution, at locations where the two solutions differed. These differences are truly negligible; hence, each two-dimensional Riemann solution was executed using $v_0^2 = 0$ and $v_0^2 = v_{opt}$.

Computational assessments are reported (Baker, 1982) evaluating vanishing gradient boundary conditions applied on the transverse walls of the two-dimensional inviscid flow Riemann shock tube. Figure 8.16 graphs the $M = 32 \times 6$, $k = 1$ solution at $t = 0.14154$ s, for the shock tube axis aligned parallel with the principal coordinate $x_1$. As a consequence, the entire momentum solution is carried by $m_{11}$, and $m_{21}$ was computed equal to zero to three significant digits for the Newton convergence set at $\epsilon = 0.01$. The $M = 32 \times 6$ solution agrees exactly with the comparison $M = 32$ one-dimensional $k = 1$ algorithm solution (Fig. 8.17), even to prediction of the small amplitude $2\Delta x$ waves in the zero velocity region upstream of the rarefaction wave. The solution required two iterations per step for convergence, and executed in 27 time steps. Figure 8.18 graphs the solution for the more demanding case corresponding to insertion of the diaphragm at an angle to the tube axis. This yields the algorithm requirement to predict a shock oblique to the mesh. The algorithm results exhibit comparable accuracy, and experience no difficulty in enforcing the gradient boundary for steep solution gradients oblique to the walls. A slightly enhanced "2\Delta x" trashiness occurs in the solution right rear corner, and a modest no-zero distribution for $m_{21}$ was also computed.

Figure 8.19 graphs the $M = 32 \times 6$, $k = 1$ algorithm solution for the shock tube misaligned with the principal coordinates of the momentum vector $m_1$. As a consequence, both momentum equations generate non-zero solutions. As can be seen by comparing Fig. 8.16, the shock does not appear prominent in these $m_i$ solutions (Fig. 8.19b-c). However, computation of the component of $u_i$ parallel to $\eta_1$ (cf. (8.120)), which is the coordinate parallel to the shock tube axis, confirms existence of the shock (Fig. 8.19c). Further, the predicted internal energy distribution (Fig. 8.19d) is also in good agreement with the "correct" solution (Fig. 8.15).

The final evaluation reported is for the aligned shock tube definition, reexecuted as a viscous problem at Re $\equiv 10^6$ on an $M = 32 \times 20$ uniform discretization. The corresponding momentum boundary conditions on the transverse walls are now no-slip, i.e., $m_1 = 0 = m_2$. The cold wall boundary condition was defined for the energy equation $\epsilon \equiv \epsilon_0$ in (8.7), and a vanishing normal derivative applied for the density solution (8.1). No boundary condition specifications are appropriate for $p$, $a_{ij}$, or $q_i$, since each is defined by an algebraic equation. The integration time step and Newton convergence requirement were maintained identical to those of the inviscid test case (Fig. 8.16). Figure 8.20 graphs the $k = 1$ algorithm solution at $t = 0.14154$ s.
the centrosid region, away from the influence of the no-slip wall, the viscous solution agrees essentially exactly with the comparison one-dimensional inviscid solution (Fig. 8.16). The influence of the viscous boundary conditions is clearly evident in all dependent variables. The growth of the laminar boundary layer behind the traveling shock is just visible in the solution for $m_1$. The solution for $m_2$ is quite oscillatory, but the peak value is only about 10 percent of the maximum $m_1$. The $\sigma_{12}$ shear stress solution is sharply peaked at the wall, with a steep front adjacent to the shock. The solution is non-zero only in the vicinity of the wall, and exhibits the required skew symmetries.
8.12 THREE-DIMENSIONAL FORMULATION.  
LINEAR TENSOR PRODUCT BASIS

The metric data for a three-dimensional problem definition are generated in the manner discussed for $R^3$ in Sec. 8.10. The array of nodal coordinate triples of the discretization $ Ur^3$ are denoted as \( \{XI\} \equiv \Sigma \{XI\}_e, 1 \leq e \leq M \) and 1 \( \leq l \leq 3 \). For the linear tensor product basis \( \{W^l\} \), the finite-element domain $R^3$ possesses only vertex nodes \( \{XI\}_e \equiv \{XI, YI, ZI, 1 \leq l \leq 8\} \). For the definition given in (8.115), the elements on the trace of \( [\partial \eta_i / \partial x_l] \), evaluated at the centroid \( \{\eta_i\} = 0, 1 \leq l \leq 3 \) of $R^3$ are

\[
\{ETA1\}_e^2 = \frac{1}{2} \{Y41Z51, Y32Z73, Y41Z84, Y85Z51, Y76Z73, Y85Z84\}_e
\]

\[
\{ETA2\}_e^2 = \frac{1}{4} \{X21Z51, X21Z62, X34Z73, X34Z84, X65Z51, X65Z62, X78Z73, X78Z84\}_e
\]

\[
\{ETA3\}_e^2 = \frac{1}{4} \{X21Y41, X21Y32, X34Y32, X34Y41, X65Y85, X65Y85, X78Y76, X78Y85\}_e
\]

(8.128)

For a rectangular domain, \( \{ETAKJ\}_e = \{0\} \) for $K \neq J$, and the notation in (8.128) is defined as, for example,

\[
Y41Z51 = (Y4 - Y1)(Z5 - Z1)
\]

(8.129)

In the same notation, the elements of \( \{DET\}_e \) evaluated at the origin of the domain $R^3$ are

\[
\{DET\}_e^2 = \frac{1}{2} \{X21Y41Z51, X21Y32Z62, X34Y32Z73, X34Y41Z84, X65Y85Z51, X65Y76Z62, X78Y76Z73, X78Y85Z84\}_e
\]

(8.130)

In the instance of $R^3$ being a rectangular parallelepiped, then \( \{DET\}_e = \Delta_e \{1\}/64 \), where \( \Delta_e \) is the element volume. For (8.128)-(8.130), nodes 1-4 are defined in the lower plane (see Fig. 8.1b), with node 1 in the lower left corner. Nodes 5-8 are also ordered counterclockwise, in the upper plane, with node 5 above node 1.

For the tensor product Jacobians (8.74)-(8.79),

\[
\{DETK\}_e^2 = \frac{1}{2} \sqrt{(XKR - XKL)^2 + (YKR - YKL)^2 + (ZKR - ZKL)^2}
\]

(8.131)

where $R$ and $L$ denote right and left, respectively, for the specific $n_e$ alignment. As occurred for $R^2$, the components of \( \{ETAJK\}_e \) are the direction cosines of the local affine transformation between $x_l$ and $\eta_i, 1 \leq (i, j) \leq 3$ [see (8.123)]. The definition for \( \{UBARK\}_e \) is also unchanged from (8.124).

No consequential solutions using the three-dimensional algorithm are reported in the literature. However, Baker (1982) does document the three-dimensional tensor product Jacobian construction for the three-dimensional equivalent of the Riemann shock tube.

8.13 CLOSURE

This concludes the derivation and examination of the theoretical and practical aspects of finite-element computational fluid mechanics. The reader has—it is hoped—benefited from the experience. It should be quite obvious that this science is in its infancy, with regard to application to the complete Navier-Stokes equations. Nevertheless, the theoretical structures are rich in classical mechanics, and the formational procedures have fully utilized calculus and vector field theory. Hopefully, the robustness of this methodology will prompt continued refinement and examination of its application to real-world problem classes in computational fluid mechanics.

REFERENCES


Efficient utilization of the uniformity of the finite-element algorithmic procedure is enhanced by cardinal basis functions and a unified notation. These concepts, which are extendible to multidimensional space, are best introduced on one-dimensional space. Figure A.1 shows a one-dimensional element with vertex nodes 1 and 2. Linear interpolation of the distribution of a variable \( q(x) \) over \( I_e^2 \) is

\[
q_e(x) = a + b \frac{x}{\Delta e}
\]  

(A.1)

The coefficients in (A.1) are easily reexpressed in terms of the nodal values \( Q_i \) of \( q_e \) as

\[
q_e(x) = Q_1 \left(1 - \frac{x}{\Delta e}\right) + Q_2 \left(\frac{x}{\Delta e}\right)
\]

(A.2)

where the origin of \( \bar{x} \) is at node 1. Rewriting Eq. (A.2) as a matrix inner product produces the linear one-dimensional cardinal basis \( \{N_1(\bar{x})\} \):

\[
\{N_1(\bar{x})\} \equiv \begin{bmatrix} 1 - \frac{x}{\Delta e} \\
\frac{x}{\Delta e} \end{bmatrix}
\]

(A.3)

Equation (A.3) defines the linearly dependent, normalized natural coordinate system \( \xi_f \) for a one-dimensional space, that is, \( \xi_f = \{N_1(\bar{x})\} \). The \( \xi_f \) system is also
Finite Element Computational Fluid Mechanics

A. J. Baker
Department of Engineering Science and Mechanics
The University of Tennessee, Knoxville

Minkowycz and E. M. Sparrow, Editors